MASTER EN SCIENCES MATHÉMATIQUES

Implémentation d'une solution simulant la dynamique des mariages au sein des communes belges

Collot, Alice

Award date:
2018

Awarding institution:
Universite de Namur

UNIVERSITE DE NAMUR

Faculté des Sciences

Implémentation d'une solution simulant la dynamique des mariages au sein des communes belges

Promoteur : CARLETTI, T.
Co-promoteur : CORNELIS, E.

Mémoire présenté pour l'obtention du grade académique
de master en "sciences mathématiques, à finalité spécialisée en Data Science"

Alice COLLOT

Juin 2018

Résumé

Pour améliorer les besoins de leur population, les politiques ou entreprises proposent des projets, lois qui offrent des solutions dans un futur plus ou moins proche. Or, ces décisions, avant d'être prises, doivent être réfléchies en s'appuyant sur une analyse robuste et complète. Le domaine des mathématiques est souvent sollicité, notamment grâce à ses innovations au niveau des outils de modélisation. Il est maintenant possible de simuler les comportements et les caractéristiques d'une population réelle. Ces populations synthétiques implémentées doivent être statistiquement proches de la réalité. Elles peuvent comprendre plusieurs dynamiques d'évolution distinctes comme le module des naissances, celui des décès ou encore le vieillissement des individus, qui simulent les processus démographiques réels.

Notre mémoire poursuit donc l'objectif de modéliser une dynamique particulière de la population : celle des mariages. Cette simulation complexe se concrétise en deux modules indépendants : la construction du marché des mariages et ensuite la formation du couplage. Plusieurs méthodes sont implémentées et analysées pour chaque partie. Nous tentons de dégager les algorithmes les plus adaptés à chaque étape et de les généraliser pour qu'ils soient facilement intégrables dans un projet de population synthétique.

Mots clés : choix discrets, couplage, marché du mariage, mariages, micro-modélisation, recuit simulé, réseau de neurones, simulation, statistiques.

Abstract

To improve people's needs, politics and companies offer plans or laws which lead to future solutions. However, these undertakings have to be thought before taken, a full and solid analysis has to be realized. To achieve this study, mathematics are often used because of their progress in modelisation tools. It's now possible to replicate the behaviour and the features of the actual population. This synthetic population is not identical to the real one but has to reproduce different statistical distributions of the actual population in order to be a proper modelization. It includes various dynamics of evolution, such as the aging of people, the modeling of births or the simulation of deaths. These dynamics simulate real lifecourse process.

The primary goal of this work is the modeling of a specific population dynamic : marriages. This complicated process involves two separate parts. First, the marriage market is built with all of the single people who desire to get married. Then, the matching is performed. Several models are implemented and studied for each of the two sections. In the end of each part, we conclude on the more suitable algorithm. Furthermore, we want these models to be general in order to be inserted in current synthetic population projects.

Keywords : discrete choice, marriages, marriage market, matching, micro-modeling, neural network, simulated annealing, simulation, statistics.

Remerciements

En préambule à ce mémoire, je tiens à remercier sincèrement toutes les personnes qui ont contribué de près ou de loin à son élaboration.

Mes remerciements vont en premier lieu aux personnes qui ont accepté de me superviser et me suivre dans ce travail de recherche, je cite les Docteurs Timoteo Carletti et Eric Cornelis. En tant que promoteurs, ils m'ont partagé leurs conseils et m'ont apporté plusieurs pistes de réflexion pour poursuivre et atteindre mon objectif. Je tiens ensuite à remercier chaleureusement Morgane Dumont pour son encadrement, sa disponibilité mais surtout pour son son aide tout au long de ce travail. Je remercie aussi le groupe DEMO de l'UCL qui a gracieusement mis à disposition des données sur la population belge dans le cadre du projet VBIH financé par la Wallonie et dont j'ai pu profiter pour une partie cruciale de mon mémoire.

Je remercie ensuite de tout cœur Maman qui a passé de très longues heures à lire et relire ce travail. Ses conseils, son aide et son soutien sont inestimables. Ces remerciements peuvent aussi être adressés à DD. Je ne compte plus le nombre d'heures passées au téléphone avec elle dès qu'un doute ou une question me venait. Merci à elles deux pour leurs encouragements et leur grande estime à mon égard.

Je n'oublie pas toute ma famille, Nala, Papa, Mathias, Marraine, qui m'ont soutenue mais surtout supportée durant ces quelques mois de travail. Leurs petits services ne sont pas oubliés. Je remercie aussi mes correcteurs, Maminou, Jean-Claude et Fanfi, qui m'ont fourni des remarques avisées.

Je tiens ensuite à remercier Yves-Henri et Sabine qui m'ont offert l'opportunité de travailler avec eux sur un projet très enrichissant. Outre leur reconnaissance, ils m'ont apporté leurs connaissances mais aussi un soutien important.

Je remercie finalement Mat pour nos pauses cupcakes les lundis mais surtout pour sa correction amusante. Avec les autres filles, elles m'ont permis d'oublier quelques heures le travail à accomplir lors de ces rares soirées sur Namur.

Merci à tous et toutes.

Table des matières

Table des figures

Liste des tableaux

Introduction

Si l'Humanité a évolué depuis sa création, c'est qu'elle a pu résoudre les difficultés auxquelles elle était confrontée. Ces problèmes surviennent encore actuellement et proviennent de divers besoins, dans différents domaines comme l'économie, la politique ou encore le médical. Les Hommes veulent constamment améliorer leur confort tout en jouissant d'une longue vie en bonne santé. C'est pour poursuivre cet objectif que naissent les lois, les politiques ou encore les projets. Or, pour soumettre des solutions adéquates et optimales aux problèmes rencontrés, ces propositions doivent s'appuyer sur des analyses consistantes. Très souvent, le domaine mathématique est sollicité, notamment grâce à ses méthodes et outils de modélisation. Les techniques de simulation mathématique actuellement disponibles ont atteint un niveau de développement et de maturité tel qu'elles permettent la modélisation de populations réelles, appelées populations synthétiques. Ces dernières doivent être statistiquement proches de la réalité pour pouvoir obtenir des résultats corrects et précis. De nombreux décideurs voient ainsi un intérêt à disposer d'une population virtuelle afin d'agir au mieux.

Or, les décisions prises actuellement ont pour objectif de résoudre les difficultés qui se manifestent ou menacent de survenir dans un avenir plus ou moins lointain. En effet, elles ne peuvent rendre leurs effets immédiatement et auront, en conséquence, un impact sur la postérité du pays ou de la zone géographique concerné ; ce sont les besoins futurs de la population qui doivent être connus. Disposer d'une population virtuelle n'est pas suffisant, celle-ci doit pouvoir se développer temporellement afin de connaître ses caractéristiques à venir. Pour ce faire, il faut simuler l'évolution de ses attributs, comme l'âge et la qualification de ses individus, ou encore les naissances et les décès. Les relations entre les personnes ne doivent pas non plus être oubliées. Parmi celles-ci, nous recensons le divorce ou encore la mise en couple, le mariage. C'est sur cette dernière dynamique que s'attarde notre mémoire. La finalité de ce travail est donc l'implémentation d'un algorithme qui simule le processus des mariages au sein de la population belge.

Tout d'abord, bien que nous ne mentionnons que le terme mariage, nos résultats et analyses restent valables pour une dynamique similaire, la mise en couple. Ce sont tous deux des processus de sélection de partenaires et certains modèles tirés de la littérature n'envisagent que ce dernier module comme The SimMunicipality Model [HUE2012]. Ensuite, le mariage est un processus ayant des effets importants sur la population synthétique dans son ensemble mais en même temps reste une dynamique fort complexe. Détaillons ces deux points importants. Tout d'abord, la création et la séparation des couples sont les premiers facteurs expliquant la mobilité résidentielle selon l'analyse des données provenant d'une enquête française [DEB2005,DEB2006]. Ensuite, plusieurs dynamiques sont influencées par ce processus. Prenons l'exemple de la natalité. Une femme mariée possède une plus grande probabilité d'enfanter qu'une femme célibataire. Enfin, le premier chapitre appuie l'importance de la modélisation des mariages. En effet, plusieurs populations synthétiques provenant de la littérature ne tiennent pas compte du statut « en couple » au sein des cohabitations, par exemple DYNACAN au Canada [MOR2000]. Autrement dit, si deux individus ne sont pas mariés au sens juridique, ils ne sont pas considérés comme formant un couple et n'ont donc pas la possibilité de concevoir des enfants dans le modèle implémenté. Or, dans la réalité, il n'est pas nécessaire de se marier pour procréer. Dans ces populations,

nous notons l'importance de posséder un module des mariages consistant puisque les époux sont les seules personnes qui peuvent influencer positivement la balance naturelle en engendrant des descendants. Si l'algorithme produit peu de mariages, la population virtuelle s'éteindra après quelques années.

En modélisant la dynamique des mariages, nous faisons face à plusieurs problèmes ne se présentant pas dans d'autres processus. Tout d'abord, non pas un mais deux individus sont concernés par cet algorithme. En effet, il faut coupler les partenaires de manière à respecter les attentes des deux parties. Les divorces ne rencontrent pas cette complication puisque si un époux décide de divorcer, il ne laisse pas l'opportunité à son conjoint de décider d'une autre issue. La deuxième difficulté relevée est le besoin d'une grande quantité de données sur les individus mariés de la société étudiée afin de modéliser de manière correcte et précise le processus de couplage. Nous devons posséder des informations sur ces personnes avant et après leur mariage. Néanmoins, la plupart des enquêtes offrant ce type de données contiennent peu de variables. De plus, la dynamique dépend de facteurs personnels comme les attentes sur le partenaire idéal. Ces données ne sont évidemment pas disponibles. Un dernier problème concerne la modélisation de la dynamique. En micro-simulation, nous allons d'abord déterminer les individus voulant se marier et ensuite, leur trouver un partenaire. Malheureusement, cela ne se concrétise pas souvent de cette manière en réalité. Les époux habitent généralement ensemble quelques années avant d'envisager le mariage. Peu de personnes décident de se marier au cours d'une année sans avoir, au préalable, un partenaire potentiel. Cette constatation débouche sur une dernière problématique, celle des personnes vivant en cohabitation. En effet, si ceux-ci désirent se marier, leur statut de cohabitants légaux est simplement modifié en ménage marié. Ces individus ne doivent alors pas être considérés dans l'ensemble des célibataires disponibles. Dans ce mémoire, nous essayons de surpasser ces difficultés et de proposer une modélisation la plus réaliste et générale possible. Cette dernière caractéristique est essentielle pour intégrer notre module dans une population synthétique existante.

Pour offrir la solution implémentée attendue, nous commençons par étudier la littérature concernant cette dynamique. Le premier chapitre débute avec la définition de quelques concepts théoriques importants à la compréhension de ce travail pour ensuite parcourir une large palette de modélisations existantes afin d'en extraire les idées principales ou innovantes. Le deuxième comprend deux analyses statistiques sur les données utilisées par la suite. La première étudie l'influence de certaines variables sur la décision de se marier et la seconde se concentre sur les informations prises en compte lors du choix du partenaire. Ensuite, nous tentons de construire le marché des mariages, l'ensemble des individus désireux de se marier l'année considérée. Nous exploitons deux théories différentes que nous explicitons au préalable : les choix discrets et les réseaux de neurones. Le chapitre subséquent essaye, lui, de former les couples au sein de ce marché. Nous commençons par définir les préférences des hommes envers chaque femme du marché des mariages. Le couplage est ensuite réalisé par deux méthodes différentes. Avant de conclure, nous désirons proposer une perspective pour notre solution. Nous voulons l'intégrer dans une population synthétique existante, celle du projet VirtualBelgium in Health (VBIH). Par manque de temps, ce chapitre contient la description complète du projet ainsi que de ses modules mais seulement quelques pistes pour l'adaptation de nos modèles au sein de VBIH.

Chapitre 1

État de l'art

Pour pouvoir implémenter une solution consistante, simulant le processus des mariages dans une population synthétique ainsi que pour éviter de gaspiller des ressources temporelles précieuses, il est utile d'analyser la littérature à propos de notre sujet. En effet, il est intéressant d'explorer plusieurs solutions à notre problématique mais certaines voies sont conseillées puisque le sujet abordé n'est pas récent et que beaucoup de recherches s'y sont consacrées. Ce chapitre a donc, comme objectif principal, d'étudier brièvement la littérature dédiée à la modélisation de la dynamique du mariage. Nous avons préféré explorer plusieurs articles, livres, travaux, ... plutôt que de nous focaliser sur l'une ou l'autre proposition dans le détail. Nous voulons effectivement implémenter une solution propre tout en nous appuyant sur des bases existantes.

Pour atteindre l'objectif de ce chapitre, nous commençons, dans un premier temps, par développer des concepts théoriques importants pour la compréhension de notre mémoire, notamment en caractérisant plusieurs attributs qu'un modèle de micro-simulation peut posséder. Ce type de modèles est défini au préalable. Ces caractéristiques sont, en effet, primordiales puisqu'elles influencent directement la dynamique des mariages. Dans un second temps, nous élaborons un rapide historique des modèles de micro-simulation se focalisant sur notre dynamique. Nous donnons leurs caractéristiques essentielles en vue de retirer plusieurs solutions pour la suite de notre travail. La dernière section se concentre sur ce dernier point : résumer soit les idées innovantes de ces modèles, soit leurs points communs. Nous voulons, en effet, posséder des bases solides pour débuter notre travail. Dans ce chapitre, nous avons principalement utilisé les ressources [COR2015,HUE2013,NAM2014,PIC2014,GEA2013,MOR2010, ZAD2001, PEI2017, MUD2015, ZIN2012] pour nous aider.

Sommaire

1.1 Concepts théoriques

Cette section comporte deux parties : la première concerne les systèmes multi-agents et la seconde, les modèles de micro-simulation. Bien que notre intérêt se porte sur ces derniers, nous trouvons important de donner un contexte d'utilité de ces modèles. En effet, les populations synthétiques implémentées sont généralement construites comme des systèmes multi-agents et les dynamiques d'évolution, comme le mariage, peuvent être simulées grâce à ces modèles de micro-simulation.

1.1.1 Systèmes multi-agents

Cette partie théorique repose uniquement sur la ressource [CHA2001]. Avant de définir les systèmes multi-agents, nous devons introduire une notion importante, celle d'agent. Ce mot est un nom assez commun, populaire qui est utilisé dans diverses disciplines comme l'intelligence artificielle ou encore la psychologie. Étant donné son usage dans des situations variées, nous trouvons une quantité importante de définitions le concernant. Bien que, selon le domaine d'application, des caractéristiques particulières se détachent, toutes ces définitions possèdent un fond commun. Commençons par une première description [1] :

> *Un agent est une entité autonome, réelle ou abstraite, qui est capable d'agir sur elle-même et sur son environnement, qui, dans un univers multi-agents, peut communiquer avec d'autres agents, et dont le comportement est la conséquence de ses observations, de ses connaissances et des interactions avec les autres agents.*

Nous dégageons de cette explication des concepts importants comme l'autonomie, la communication entre les agents ou encore la prise d'actions. Étant donné que certaines notions capitales manquent, une autre définition [2] s'impose :

> *Un agent est un système informatique, situé dans un environnement, et qui agit d'une façon autonome et flexible pour atteindre les objectifs pour lesquels il a été conçu.*

Selon les auteurs de l'article sur lequel nous nous reposons, quelques notions doivent être précisées. Un agent est situé dans un environnement quand il est capable d'interagir avec ce dernier au moyen des informations qu'il perçoit de celui-ci. Ensuite, un agent est autonome si ses actions sont sous son contrôle et se déroulent sans intervention extérieure, provenant d'humains ou d'autres agents. Il doit aussi être capable de contrôler son état interne. Enfin, définir l'agissement de manière flexible d'un agent est un peu complexe. En effet, il doit pouvoir répondre à un problème donné à temps, être proactif et opportuniste, être capable d'interagir avec d'autres agents. Cette dernière exigence permet à l'agent d'obtenir de l'aide pour accomplir sa tâche ou alors d'en apporter à ses homologues. Ces diverses propriétés possèdent plusieurs niveaux d'importance selon le domaine d'application et parfois d'autres caractéristiques sont requises.

Il faut savoir qu'un agent pris individuellement offre des possibilités limitées. En effet, posséder un ensemble d'agents travaillant sur des tâches communes semble plus intéressant. Ce groupement d'agents forme alors un système multi-agents. Définissons ce concept.

1. Définition provenant de la ressource Ferber, J., *Les systèmes multi-agents, vers une intelligence collective*, InterEditions, 1995.

2. Définition provenant de la ressource Jennings, N., Wooldridge, M., Sycara, K., *A roadmap of agent research and development*, Int Journal of Autonomous Agents and Multi-Agent Systems, 1(1), 7- 38, 1998.

Un système multi-agents, SMA, est composé de plusieurs agents qui interagissent entre eux. C'est de plus un système distribué. Les interactions inter-agents sont de l'ordre de la coopération, de la coexistence ou alors de la concurrence. De plus, cette communication, quelle qu'en soit la forme, peut se dérouler de plusieurs manières : en interaction directe, via un agent intermédiaire ou encore par une modification de l'environnement dans lequel l'ensemble des agents se trouve. Une population synthétique est alors un système multi-agents où ceux-ci représentent les individus de la population. Ils interagissent entre eux notamment par les liens familiaux, le mariage ou encore le divorce.

Un SMA possède d'autres caractéristiques :

- tout agent bénéficie seulement d'un point de vue partiel puisqu'il dispose d'informations et de capacités de résolution limitées,
- il n'existe pas de contrôle global pour le SMA,
- les données ne sont pas centralisées,
- le calcul n'est pas synchrone.

Le système multi-agents est alors une approche innovante pour la conception, l'implémentation mais aussi l'analyse de systèmes informatiques complexes. Il est idéal pour modéliser des problématiques ayant plusieurs méthodes de résolution. Un des avantages non négligeables de cette théorie est son caractère distribué qui offre la modularité, le parallélisme et par conséquence, la vitesse et enfin la fiabilité des informations, grâce à la redondance.

Les auteurs de la ressource utilisée pour cette section pensent que les SMA vont prendre de l'ampleur en avançant dans les années. Ils pourraient alors représenter un nouveau paradigme pour la programmation au même titre que la programmation orientée objet. Les SMA pourraient se dénommer, ultérieurement, programmation orientée agent. Il faut être vigilant car malgré la ressemblance entre les termes agent et objet, ce sont des concepts différents. Tous deux encapsulent leurs données, leur état interne et peuvent alors effectuer des actions sur ceux-ci. Ils sont, de surcroit, aptes à communiquer des messages via leurs propres méthodes. Une différence principale vient de l'autonomie des agents par rapport aux objets. Pour pouvoir effectuer une méthode d'un objet, celle-ci doit effectivement être invoquée par un autre objet. Au contraire, quand un agent perçoit une requête, il peut décider de son propre gré d'agir ou non. Une autre dissemblance relevée concerne la flexibilité des agents que nous ne retrouvons pas chez les objets. Enfin, la dernière différence porte sur la notion de contrôle dans ces systèmes. Pour celui orienté objet, une seule source de contrôle est observée tandis que chaque agent est considéré comme une source de contrôle à l'intérieur du système.

Ces explications ne sont pas vaines puisque notre mémoire se focalise sur une interaction particulière entre les agents : le mariage. Nous cherchons à modéliser cette communication. C'est pourquoi la section suivante est essentielle. Elle donne les informations nécessaires à la compréhension des modèles de micro-simulation, permettant de simuler notre dynamique.

1.1.2 Modèles de micro-simulation

Afin de reproduire le comportement d'individus dans une société, il est préférable d'utiliser un modèle dynamique de micro-simulation. Ce dernier permet de modéliser les processus d'un système, d'une population ou encore d'une société en simulant le comportement de leurs micro-unités, principalement des individus ou ménages. Ce sont des techniques qui offrent des possibilités d'analyse de projections de populations, d'impacts ou changements de politiques. C'est un modèle assez riche qui se distingue

d'autres modèles travaillant sur des groupes puisqu'il tient compte de l'hétérogénéité et de la diversité de la population concernée. La micro-simulation permet, entre autres, d'étudier l'impact temporel d'évènements discrets au-delà du court terme puisque l'évolution de ses micro-unités peut être réalisée sur plusieurs dizaines d'années. Il existe tout de même une difficulté à utiliser ce genre de modèle : des données sociales ou économiques doivent être générées, souvent provenant d'enquêtes.

Dans le cadre de ce mémoire, seuls les modèles démographiques de micro-simulation nous intéressent. Ils simulent les cycles de vie individuels grâce à une suite d'évènements démographiques (naissance, mort, mariage, ...) et au temps passé entre ceux-ci. Ces évènements peuvent se classer en trois catégories : l'évolution de la population (naissance, mort, migration), le développement des ménages (départ des enfants, cohabitation, mariage, divorce, séparation) et un troisième ensemble moins spécifique, qui concerne l'éducation, la santé et le travail. Notons qu'une description réaliste des dynamiques nécessite la considération des liens entre individus. En effet, au sein de toute population, ils interagissent constamment entre eux, que ce soit à travers de liens familiaux ou encore par le mariage.

Il existe principalement deux types de modèles démographiques de simulation de micro-unités : les MSM (MicroSimulation Models ou, en français, modèles de micro-simulation) et les ABM (Agent-Based Models, étant les modèles multi-agents). Tous deux poursuivent des objectifs différents et possèdent leurs propres avantages.

Les MSM dépendent fortement des données et en exigent souvent une grande quantité. Leur but principal est de prédire et d'analyser les impacts de certaines politiques sur la population concernée. Les agents des MSM possèdent des états dont un changement se produit avec une probabilité dépendant de leurs attributs. Beaucoup d'états sont autorisés sans pour autant posséder une estimation de leur probabilité. Par exemple, la probabilité de devenir belle-mère est souvent inconnue, pourtant cet état est présent dans la population. Les MSM tiennent alors compte de l'hétérogénéité de la population grâce aux caractéristiques spécifiques des individus. Ils peuvent, par conséquent, être utilisés pour modéliser l'évolution démographique d'une population et être le point de départ pour des modules plus compliqués.

Les ABM, quant à eux, ont pour cible de décrire les interactions entre les individus afin d'analyser l'émergence de caractéristiques globales. C'est donc un outil de modélisation comportementale. Ces modèles reposent sur le principe que chaque agent possède des règles sur lesquelles il s'appuie pour interagir avec d'autres individus et l'environnement. Aussi appelés individual-based models (IBM), ils sont évidemment en relation avec la section précédente concernant les systèmes multi-agents. L'objectif poursuivi par les ABM concerne l'analyse ou la découverte de particularités dans le comportement collectif alors que les SMA se focalisent uniquement sur l'architecture d'un ensemble d'agents.

Certaines caractéristiques doivent encore être déterminées afin d'élaborer un algorithme acceptable pour la dynamique du mariage. En micro-simulation démographique, nous pouvons relever trois approches pour modéliser le processus de couplage au sein d'une population. Le modèle peut être ouvert, fermé ou encore ignorer cette procédure. Expliquons brièvement ces trois idées divergentes.

Nous pouvons effectivement ignorer le processus du couplage, entrainant une indépendance entre les cycles de vie des individus. Quand un célibataire de la population réalise la dynamique du mariage, aucun partenaire ne lui est attribué ni créé. La population est démunie de couple synthétique. Une conséquence directe est l'incapacité d'étudier les dépendances entre la vie d'une femme et de son mari. Cette idée est envisageable uniquement lors de modèles ne tenant pas compte de la formation de ménages. Elle possède certains avantages puisqu'aucune donnée concernant les couples de la population

n'est nécessaire ou encore car les règles de couplage ne doivent pas être définies. Néanmoins, d'importants points négatifs sont relevés : la composition de la population n'est pas étudiée ou encore aucune analyse sur les liens familiaux ne peut être réalisée. Si nous voulons approfondir la corrélation entre le taux de natalité et les relations mère-fille, il n'est pas possible de le faire ici.

Dans le cas d'un modèle ouvert, quand un individu se marie, son partenaire est créé avec des caractéristiques appropriées. Cette personne n'est donc pas un agent de la population synthétique générée. Les attributs des partenaires ajoutés sont produits pour que la distribution des couples soit similaire à la réalité. L'implémentation de ce modèle est assez simpliste et demande peu de ressources informatiques puisqu'il n'est pas nécessaire d'identifier un partenaire dans la population. Cette structure implique que les dynamiques pour chaque individu peuvent être évoluées indépendamment des autres personnes dans la population. Bien que ce modèle soit peu complexe, il possède quelques principaux désavantages. Tout d'abord, il est possible que les individus générés ne modélisent pas correctement les mariages puisque des personnes adéquates sont conçues au besoin. Or, en réalité, les individus doivent choisir un partenaire existant et par conséquent, ne pas se marier avec l'homme ou la femme idéal mais bien relâcher certaines attentes. Ensuite, son interprétation reste difficile puisqu'il ne modélise pas le couplage en réalité. Une difficulté liée à ce constat est l'historique manquant des partenaires créés. Au moment où ceux-ci sont ajoutés dans la population synthétique, nous n'avons aucune idée de leur parcours de vie.

À l'inverse, en situation de modèle fermé, les agents n'entrent dans la population virtuelle que par naissance, ou par immigration et ne la quittent que par mort ou émigration. Cela signifie que lorsqu'un célibataire désire se marier, un compagnon doit être trouvé parmi les individus existants. La plupart des modèles fermés réalisent la dynamique des mariages en deux étapes. Tout d'abord, les individus célibataires voulant se marier sont assemblés dans un groupe fictif, appelé marché du mariage. D'un point de vue technique, ce marché est considéré comme une liste triée d'individus. Le critère de tri est généralement l'âge des personnes. Un attention spéciale doit être portée sur les cohabitants légaux désirant se marier. Ceux-ci souhaitent changer leur statut et non pas trouver un individu parmi la population puisqu'ils possèdent déjà un partenaire. Ils ne doivent, en conséquence, pas être considérés dans ce marché. La seconde étape est la réalisation du couplage (ou matching, en anglais) des agents deux à deux. Pour déterminer la qualité d'un couplage particulier, une mesure de compatibilité doit être définie. Cette dernière quantifie la valeur d'un matching en se basant sur les attributs des partenaires. Concluons sur l'avantage et l'inconvénient d'un modèle fermé. Ce modèle est plus complexe à implémenter mais s'interprète plus facilement. Il permet d'étudier les effets du processus de couplage sur la composition de la population.

À présent, regardons à l'évolution temporelle du cycle de vie d'un individu dans un modèle de microsimulation. Tant l'évolution de l'âge que la dynamique du calendrier doivent être modélisées et peuvent toutes deux être une période temporelle discrète ou continue. Donnons quelques explications et caractéristiques sur ces deux possibilités. Dans un modèle en temps discret, l'axe temporel est discrétisé, souvent en années ou en mois. À chaque étape, les attributs des individus sont mis à jour. Les modèles se basant sur un temps continu possèdent une échelle temporelle continue où les évènements démographiques peuvent se produire à tout moment. C'est le choix théorique optimal car ce modèle correspond mieux à la réalité. En outre, il est plus efficace car les caractéristiques des individus sont mises à jour uniquement quand un évènement se produit. Malheureusement, en temps continu, il est difficile de faire évoluer la dynamique du mariage. En effet, les évènements se produisant à des moments très précis, deux individus n'expérimentent jamais le mariage au même instant. Une solution plausible est l'utilisation d'un modèle pseudo-continu. C'est un modèle en temps discret possédant une période très petite. Ses résultats sont similaires à ceux du continu.

1.2 Étude de la littérature

Voici une liste non-exhaustive de modèles de micro-simulation dynamiques traitant exclusivement ou en partie du module lié aux mariages. Nous donnons des généralités mais aussi quelques informations spécifiques sur chaque modèle. Nous ne nous étendons pas sur chaque modèle car notre objectif est de soutirer quelques caractéristiques utiles pour débuter notre solution.

Le premier modèle intéressant relevé est DYNASIM2, construit par Orcutt en 1976 [ORC1976]. C'est un modèle fermé, en temps discret, qui possède une dynamique des mariages pour les individus âgés de 18 ans ou plus. Les caractéristiques qu'il prend en compte pour simuler les mariages sont les suivantes : âge, race, genre, ancien statut marital, revenu, éducation, région, semaines travaillées, salaire horaire, revenu actif, confort de vie, allocation chômage.

Fin des années 80, au Canada, est apparu le modèle LifePaths [SPI2013]. Il se distingue de ses semblables par trois principales caractéristiques : c'est un modèle ouvert, qui opère en temps continu et qui utilise une base de données synthétique créée en utilisant différentes cohortes de naissances se chevauchant. Le mariage fait partie du module démographique au même titre que les divorces.

CORSIM [CAL1997] est apparu en 1990 aux États-Unis. C'est un modèle fermé en temps discret. La dynamique du mariage est divisée en deux sous-modules. Tout d'abord, les hommes et les femmes célibataires entrent dans un groupe pour se marier. Ensuite, le marché des mariages débute et essaye de mettre en couple ces personnes. Pour l'entrée des individus dans le marché des mariages, vingt groupes sont formés grâce à un modèle logistique. Ces groupes sont basés sur la race, le statut d'éducation, le genre et le nombre de semaines travaillées.

Le modèle dynamique du NATSEM (National Centre for Social and Economic Modelling), DYNAMOD ou DYNAMOD-2 [KIN1999], date de 1999-2000 en Australie. Il est fermé et en temps pseudo-continu. Les objectifs de ce modèle sont de projeter les caractéristiques de la population dans 50 ans et de fournir des informations sur certaines politiques concernant, notamment, l'éducation et le salaire des étudiants. La dynamique des mariages évolue chaque mois. Le modèle utilise des fonctions de survie pour simuler ces mariages au sein de la population. Ces fonctions prédisent le moment de réalisation hypothétique d'un évènement avant son apparition.

Au Canada, le modèle DYNACAN a été conçu en 2000 [MOR2000] pour étudier les conséquences des changements dans le plan de pension canadien (CPP). Il utilise, pour la procédure de couplage, les attributs : âge, genre, éducation, salaire, présence d'enfants, ancien statut marital. Le modèle est fermé et utilise un temps discret pour ses simulations.

Un modèle a été proposé en 2007 par des chercheurs d'Italie et d'Autriche nommé The Wedding Ring [BIL2007]. Il est principalement fondé sur le concept de pression sociale. Les auteurs pensent que l'aspiration au mariage d'un individu dépend fortement de la proportion de couples dans son réseau social. Le modèle considère deux éléments primordiaux qui opèrent entre les modèles de mariage et la structure sociale : la disponibilité des partenaires et le désir de se marier. De plus, il se démarque d'autres modèles par sa représentation de la population. En effet, elle n'appartient pas à un réseau social mais plutôt à un espace en deux dimensions, étant la localisation des individus et leur âge. La localisation est contenue en une seule dimension car les auteurs supposent que les individus évoluent dans un cercle. Leur domicile ou encore leur lieu de travail se situe donc en un point de la circonférence de ce cercle. Le nom Wedding Ring provient de cette représentation. Ce modèle se classe dans les ABM.

Hills et Todd [HIL2008] ont développé aux États-Unis en 2008 le modèle MADAM (Marriage and Divorce Annealing Model) dont l'objectif est de simuler les divorces et les processus non-concurrentiels de sélection de partenaires. C'est un modèle assez intéressant puisqu'il veut prédire les mariages dans différentes cultures. Son idée principale est la suivante : les individus cherchent des partenaires semblables à eux et relâchent leurs attentes en vieillissant. Pour mesurer la similarité, il utilise des traits pertinents, N au total. Chaque personne choisit un ensemble de k caractéristiques qui le représentent où $k < N$. Les auteurs pensent que partager des attributs semblables augmente l'attrait entre deux agents. Ce modèle émet comme hypothèse qu'un individu rencontre au maximum quatre partenaires potentiels du genre opposé sur un an. Celui-ci décide de se marier dès qu'il rencontre une personne possédant une cote égale ou supérieure au niveau de satisfaction, noté j. Ce dernier est défini comme le nombre de traits similaires entre deux individus.

En 2010-2011, J. Barthélemy, C. Hubaux, A. Füzfa, R. Lambiotte et Ph. Toint ont implémenté un modèle dans le cadre du projet multidisciplinaire dédié aux étudiants de la finalité spécialisée du master en sciences mathématiques à l'Université de Namur. La dynamique du mariage repose sur deux choix discrets que la population doit effectuer. Tout d'abord, un individu choisit entre le célibat ou le mariage. La probabilité liée à cette décision utilise les variables âge, genre, statut socio-professionnel et historique du statut matrimonial qui regroupe le nombre de changements de statut, le statut actuel ainsi que le temps écoulé depuis le dernier changement. Le deuxième choix est prendre un compagnon avec ou sans enfant. Il se base sur l'âge et le nombre d'enfants de l'individu.

Dans le cadre du projet européen PRIMA, The SimMunicipality Model voit le jour en 2012 [HUE2012]. Il fait partie des ABM et a pour objectif d'étudier l'évolution de la démographie et de l'occupation des populations dans des zones rurales européennes. Cet algorithme ne tient pas compte de la dynamique des mariages mais possède un module pour créer des couples parmi la population. À chaque étape temporelle, chaque individu célibataire possède une certaine probabilité de chercher un compagnon. Pour trouver le partenaire idéal, un individu cherche dans un premier temps sur son lieu de résidence avant de regarder dans les municipalités voisines et sur son lieu de travail. Ce partenaire doit être célibataire et dans la même tranche d'âge. L'individu peut chercher parmi les habitants mais aussi parmi les immigrants potentiels. Quand un couple est formé, le nouveau ménage choisit la plus grande résidence pour y habiter. Ce modèle est du type fermé et possède deux étapes. Tout d'abord, les individus à mettre en couple sont sélectionnés selon certaines variables. Ensuite, ils entrent dans le marché des mariages où une procédure de couplage est réalisée selon plusieurs caractéristiques (âge, niveau d'éducation, statut socio-économique, ...). Des techniques de maximisation sont utilisées et plusieurs règles de décision sont appliquées pour les individus ne trouvant pas de partenaire (relâcher certaines contraintes au niveau des caractéristiques du compagnon ou attendre une période dans le marché des mariages).

Un MSM, MicSim, a été créé, en Allemagne, en 2012 [ZIN2012] avec une particularité : l'utilisation du temps continu. C'est une difficulté importante qui est contournée par la création d'un marché de couplage dans lequel les individus peuvent entrer et sortir quand ils le souhaitent. Dans ce marché, les individus cherchent un partenaire idéal. Pour que le couplage se concrétise, le modèle imite un processus de prise de décision. Le choix d'entrer en relation dépend de modèles logit estimés empiriquement pour la probabilité qu'une certaine femme et un certain homme se réunissent ainsi que d'un niveau d'aspiration individuel concernant un partenaire potentiel. Par conséquent, un couple est formé si une décision positive est prise et si la période de formation du couple correspond aux intervalles de temps de recherche des partenaires potentiels. Ce modèle a été utilisé pour construire une population synthétique pour les Pays-Bas.

En Afrique du Sud, un modèle multi-agents a été construit au cours de l'année 2015 [MUD2015]

pour modéliser les relations sociales et sexuelles des individus en situation réelle afin d'analyser la progression du VIH dans une communauté. Chaque agent possède des attributs : l'attractivité, l'aspiration, la durée de séduction, le désir sexuel. La formation des couples est basée sur un index de sympathie calculé en utilisant l'âge, l'aspiration et l'attractivité des individus. Ce modèle suppose qu'il existe trois couches de réseaux d'interactions dynamiques. À la base, il y a le réseau d'amitié, suivi de celui sexuel et des rencontres. Au plus haut se situe celui des mariages.

En 2015, un modèle de micro-simulation est né au Luxembourg pour le projet MOEBIUS (MObilities, Environment, Behaviours Integrated in Urban Simulation) [COR2015]. Son but est d'acquérir une meilleure connaissance des interactions entre l'utilisation du territoire et la mobilité. Dans ce modèle démographique, la dynamique du mariage est l'avant-dernière à être mise à jour. Elle se déroule de la façon suivante : tous les célibataires de 15 à 60 ans sont pris en compte. Selon leur âge et leur genre, ils possèdent une certaine probabilité de se marier. Si un individu décide de se marier, dix candidats sont sélectionnés dans la liste des célibataires de genre différent et de tranche d'âge proche, entre cinq ans en moins et cinq en plus. Chaque candidat se voit attribuer un score et celui possédant le plus petit est choisi. Voici la formule pour calculer le score : différence d'âges + différence de niveaux d'éducation + différence de catégories socio-professionnelles + différence de nationalités. Finalement, quand deux personnes se marient, les deux foyers de type célibataire fusionnent pour devenir un ménage marié. Pour décider du lieu d'habitation du couple, une règle est d'application : dans 60% des cas, les individus habitent dans la maison de la femme.

1.3 Informations retenues

Pour commencer notre solution sur des bases solides, il faut tout d'abord déterminer ses caractéristiques principales. Les modèles en temps discret sont les plus répandus dans la littérature. En effet, intégrer le temps continu est fort complexe. Nous choisissons d'utiliser le temps discret pour des raisons d'adaptation. Notre module peut ainsi être intégré dans plusieurs populations synthétiques utilisant cette caractéristique pour l'ensemble de leurs dynamiques. Il n'est pas irréalisable mais fort compliqué d'insérer un algorithme en temps continu dans une population lorsque tous les attributs démographiques sont mis à jour chaque année ou chaque mois. Notre solution doit d'ailleurs être adaptée à ces deux cas de figure temporels.

Notre modèle tient compte du caractère fermé puisqu'il est populaire dans la littérature. Les individus couplés doivent alors faire partie de la population initiale. Nous ne créons pas de partenaire idéal pour les personnes voulant se marier. Cet attribut est plus complexe mais modélise mieux la réalité qu'un modèle ouvert. Nous devons être vigilants lors de l'assemblage des individus deux par deux car nous devons former le couplage optimal.

Abordons brièvement les attributs pris en compte pour implémenter la dynamique du mariage. Tout d'abord, dans plusieurs modèles, beaucoup de variables sont étudiées. Malheureusement, nous possédons peu d'informations sur les individus dans les données. Néanmoins, deux caractéristiques se démarquent par leur présence presque systématique dans la littérature, l'âge des individus et leur genre. Notre étude se focalise alors sur ces deux caractéristiques. Au niveau de l'âge, certains modèles tiennent compte des personnes âgées de 15 à 17 ans dans leur processus de couplage, d'autres conservent uniquement celles majeures. En Belgique, la législation n'autorise pas le mariage d'individus non majeurs. Néanmoins, nous considérons ces personnes dans notre solution car nos données sont groupées en classes de cinq ans. Ensuite, pour vérifier nos résultats au chapitre 4, nous pouvons émettre un premier regard critique puisqu'une idée populaire se détache dans la littérature : « Qui se ressemble s'assemble ». Les partenaires aux caractéristiques divergentes, appelés mariages extrêmes, doivent être

peu nombreux dans nos résultats.

Après avoir choisi les caractéristiques principales du modèle, il faut encore déterminer la meilleure façon pour assembler deux individus entre eux. Plusieurs modèles déjà implémentés utilisent le concept de « marché des mariages » et divisent la dynamique en deux parties distinctes. La première détermine tous les individus désirant se marier au cours d'une année ou d'un mois particulier ; la seconde réalise le couplage optimal parmi ces individus. Détaillons ces deux idées brièvement en commençant par définir le marché des mariages.

C'est un groupe contenant l'ensemble des célibataires désirant se marier pour une période - année ou mois - spécifique. Il est assez utile puisqu'il rassemble tous les partenaires éligibles. Ensuite, il faut déterminer les règles de couplage de deux individus dans cet ensemble. Deux personnes vont être mariées selon leurs caractéristiques personnelles (âge, niveau d'éducation, statut socio-économique, ...). Pour réaliser cela, deux possibilités majeures s'offrent à nous : une manière stable ou une autre stochastique. La première est connue sous le nom de « problème du mariage stable » [3]. La méthode stochastique utilise des expériences pour décider du succès des paires potentielles. Il lui faut donc une mesure de compatibilité pour déterminer la qualité du couplage. Elle répond au problème principal de la méthode stable : la production de paires « extrêmes » (par exemple, des différences d'âges trop importantes). C'est cette dernière méthode que nous décidons d'exploiter au chapitre 4.

Selon le projet Moebius, la première étape de cette démarche est réalisée grâce à des probabilités de se marier selon l'âge et le genre des individus. Ensuite, ses auteurs décident de prendre dix candidats dans la liste des célibataires d'âge proche de l'individu concerné pour seconde étape. Le partenaire est choisi en fonction de son score de ressemblance au niveau d'autres variables. Nous pensons que cette méthode n'est pas optimale, car il est possible de tirer dix conjoints improbables pour la personne considérée, mais plusieurs idées peuvent être soulignées : la ressemblance des individus mariés ou encore la réduction du marché du mariage. Ce dernier constat est aussi relevé dans le modèle MADAM qui annonce qu'un individu peut seulement rencontrer quatre partenaires potentiels par an. Pour réaliser le couplage des célibataires du marché du mariage, le modèle SimMunicipality utilise des techniques de maximisation. Cela offre une solution plus optimale puisqu'il essaye de satisfaire tous les individus. Ce modèle propose aussi des alternatives lorsqu'une personne ne trouve pas de partenaire en relâchant certaines contraintes ou en attendant une année supplémentaire dans le marché du mariage. Enfin, la solution implémentée à Namur se concentre sur des modèles de choix discrets. Cette théorie a déjà fait ses preuves dans la littérature dans des domaines économiques ou liés au transport. Nous décidons d'étudier cette possibilité.

Suite au couplage des individus célibataires désireux de se marier, des modifications doivent encore être encourues pour pouvoir insérer ce module au sein d'une population synthétique. En effet, les ménages doivent subir quelques transformations. L'habitat du couple doit être déterminé ou encore le statut du ménage modifié. Choisir le lieu d'habitation du ménage n'est pas simple. Nous avons relevé plusieurs propositions. Le couple peut emménager dans la plus grande résidence. Cette idée semble plausible puisque le ménage s'agrandit. Toutefois, elle nécessite des informations sur la taille de l'habitation des individus. Cette donnée n'est pas souvent présente dans des populations synthétiques. Le projet Moebius propose, quant à lui, de suivre cette règle : 60% du temps, le couple emménage dans le foyer féminin. Cette proposition est plus facilement réalisable.

3. Première fois mentionné à la ressource Gale, D., Shapley, L., *College Admissions and the Stability of Marriage*, Amer. Math. Monthly., 69, p. 9-14, 1962.

Chapitre 2

Étude statistique préliminaire sur la population belge

Lors de la manipulation de données, il est important de réaliser une étape préliminaire d'analyse. Celle-ci permet, en effet, d'acquérir des connaissances primaires mais aussi quelques intuitions afin de poser un regard critique sur de futurs résultats.

Les premières étapes à accomplir lors de l'utilisation d'un ensemble de données sont le nettoyage et la validation de celui-ci. Elles sont réalisées au sein de chaque code en fonction du besoin. Celles-ci sont cruciales car un problème récurrent dans l'étude de données est la présence d'erreurs, d'éléments aberrants, voire des renseignements manquants. Bien que notre ère soit dominée par l'information, les ensembles de données sont souvent désordonnés. Pour corriger cela, plusieurs méthodes numériques peuvent être appliquées. De plus, une expertise humaine est conseillée puisque certaines informations du domaine d'application ne se trouvent pas dans ces ensembles. Ces étapes demandent souvent beaucoup de temps. Toutefois, vu la provenance de nos données, nous nous y attardons peu. En effet, comme indiqué dans les remerciements, nous avons eu la chance de disposer de plusieurs jeux de données reçus dans le cadre du projet VBIH de la part du groupe de chercheurs DEMO de l'UCL. Nous exploitons plusieurs fichiers car un ensemble de données n'est pas souvent suffisamment riche pour concrétiser une étude complète.

Notre analyse statistique des données se déroule selon deux axes principaux qui correspondent aux chapitres 3 et 4 de ce mémoire. Nous avons, dans un premier temps, étudié chaque variable démographique disponible qui peut influencer un individu sur la décision de se marier au cours d'une année particulière. Le second axe d'analyse concerne les champs qui peuvent intervenir lors du choix du partenaire.

Sommaire

2.1 Étude de variables influençant la décision de se marier

La première étape importante pour implémenter les mariages dans une population virtuelle, c'est de déterminer les individus qui font le choix de se marier au cours de l'année considérée. Nous créons donc un marché des mariages suite à cette identification. Cette section vise alors à étudier les variables influençant les individus lors de leur décision, se marier ou rester célibataires.

L'ensemble des sous-sections qui suivent se base sur deux fichiers dont les variables sont explicitées à l'annexe A.1, *suivicouples.txt* et *recens2001etendu.txt*. Le premier fichier concerne tous les individus mariés en 2001 tandis que le second comprend toute la population de cette même année. Ce dernier fichier est plus volumineux mais contient moins d'informations sur les individus. Une première modification a été réalisée au niveau de l'année de mariage des couples, grâce à la variable acheciv. Afin de conserver uniquement les personnes qui se sont mariées en 2000, nous avons enlevé les individus dont le mariage a été officialisé une autre année que celle-ci. Cette date n'est pas anodine. Étant donné que nous possédons les informations de toute la population en 2001, il est intéressant d'analyser les individus ayant fait le choix de se marier l'année précédente. Ceux-ci peuvent alors être comparés aux personnes n'ayant pas pris cette décision et sont donc célibataires en 2001. L'analyse des caractéristiques démographiques permet de définir le prototype de population désireuse de se marier. Une hypothèse relative à cela a été émise : nous considérons que les individus mariés en 2000 qui ont conservé ce statut en 2001, sont toujours avec le même partenaire. Autrement dit, les personnes ne peuvent pas divorcer et se remarier sur une même année. Cette hypothèse est fondée dans notre situation puisque officialiser un divorce en Belgique peut se dérouler sur plusieurs mois, voire plusieurs années. Enfin, il est important de savoir qu'en filtrant les individus de cette manière, nous obtenons moins de couples dans nos données que le nombre réel de mariages en 2000, $37\,332$ mariages au lieu de $45\,123$ [1], soit $82,73\%$ du nombre réel. Cet écart est non-négligeable mais cette solution est la plus optimale au vu des informations que nous possédons dans les données. Pour cette section, nous considérons alors que nous ne disposons pas de la population complète mais seulement d'un échantillon représentatif des individus mariés au cours de l'année 2000.

Une seconde modification a été réalisée au niveau du fichier contenant la population de 2001. Nous voulons comparer les individus ayant décidé de se marier en 2000 avec ceux qui ont préféré rester célibataires. Nous devions donc sélectionner ces derniers parmi la population totale. Il suffit alors d'écarter les enfants (en dessous de 15 ans) et les couples mariés. Ceux-ci sont décrits dans le fichier *suivicouples.txt*. Tant pour les individus célibataires que pour ceux mariés, étant donné qu'il est plus facile de travailler avec des entiers qu'avec des variables catégorielles, nous avons modifié les diplômes et les classes d'âges en nombres consécutifs. Toutes ces modifications sont reprises, à l'annexe A.3, dans un notebook Jupyter. Tous les codes présents dans ce mémoire sont écrits en Python 3.

Chaque sous-section ci-après est construite de manière similaire : les données sont traitées sous forme de tables de contingence. Nous voulons, en effet, étudier la dépendance entre les attributs démographiques et la décision de la population de se marier ou non. Malheureusement, beaucoup de données étant confidentielles et la réception de nouvelles données étant assez longue, nous avons uniquement trois variables à notre disposition : la catégorie d'âges, le genre et le diplôme. Toutefois, ce premier attribut est majeur dans la décision de se marier au cours d'une année. Nous possédions aussi le code INS et la santé générale des individus mais nous avons délibérément choisi de les ignorer. En effet, étant donné que les couples vivent généralement ensemble avant de se marier, il est assez compliqué de retrouver leur provenance. Ensuite, la santé générale est peu précise et assez approximative. La santé

1. Nombre provenant de la ressource internet `http://www.lavenir.net/cnt/dmf20170208_00956623/combien-de-maries-et-de-divorces-en-2016-infographies`, se basant sur le SPF Économie.

d'une personne peut effectivement se dégrader rapidement comme une guérison peut survenir à tout moment. Nous aurions désiré étudier d'autres caractéristiques comme la présence ou non d'enfants ou encore l'historique des changements de statut marital. Ce sont, selon nous, des variables qui peuvent influencer une personne à franchir le cap du mariage. Le code Python utilisé pour cette étude statistique est disponible à l'annexe A.4.

2.1.1 Analyse du genre

La première analyse réalisée se concentre sur la table de contingence du genre des individus par rapport à leur décision : se marier en 2000 ou rester célibataire. Rappelons tout d'abord ce qu'est une table de contingence et pourquoi l'utiliser. C'est un tableau à double entrée qui contient les modalités de deux variables catégorielles où chaque case est remplie par le nombre d'observations possédant les deux modalités en relation. La dernière colonne et la dernière ligne sont réservées aux effectifs marginaux pour chaque modalité, soit le nombre d'observations comprenant cette caractéristique. Notons n_{ij}, le nombre d'observations possédant la modalité i pour la première variable et j, pour la seconde. Soit $n_{i.}$, l'effectif marginal pour la modalité i et $n_{.j}$, pour la modalité j relative à la seconde variable. Enfin, n représente le nombre total d'observations. Les deux variables catégorielles sont indépendantes si

$$\frac{n_{ij}}{n} = \frac{n_{i.}}{n} \times \frac{n_{.j}}{n} \quad \forall\,(i,j). \tag{2.1}$$

La table de contingence calculée est disponible à la table 2.1. Nous pouvons d'abord observer que peu de personnes se marient au cours d'une année. Ensuite, nous retrouvons une équité parfaite entre le nombre d'hommes et de femmes ayant décidé de se marier puisque les mariages homosexuels n'étaient pas légaux à cette date. C'est seulement à partir de 2003 que les mariages gays sont autorisés en Belgique. Nous avons alors affiché les informations de la table 2.1 dans un diagramme à secteurs. Celui-ci, présent à la figure 2.1, permet d'analyser la proportion de chaque genre pour les deux situations en comparaison. Le graphique indique qu'il y a peu de différences entre les hommes et les femmes ayant décidé de rester célibataires avec une constatation supérieure d'individus masculins.

Nous avons ensuite étudié l'indépendance entre les deux variables en présence dans cette sous-section afin de conclure sur un lien possible entre le genre d'un individu et la décision de se marier. Nous avons réalisé un test du χ-carré d'indépendance, qui est justifié par l'étude d'un échantillon de mariages en 2000 et non pas la population complète. La distance liée à ce test est une mesure qui quantifie l'écart par rapport à la situation d'indépendance. Elle détermine donc la distance entre les effectifs présents à la table 2.1 et les effectifs théoriques provenant d'une situation d'indépendance. Ces derniers sont obtenus grâce à l'équation 2.1 et sont observables à la table 2.2. Nous remarquons que les valeurs sont assez différentes de celles écrites précédemment. Afin d'analyser si ces différences sont significatives, calculons la distance du χ-carré, donnée par la formule

$$d^2 = \sum_{i,j} \frac{\left(n_{ij} - \frac{n_{i.}n_{.j}}{n}\right)^2}{\frac{n_{i.}n_{.j}}{n}}. \tag{2.2}$$

Statut ——————- **Genre**	Marié	Célibataire	**Tout**
Homme	37 332	3 489 336	3 526 668
Femme	37 332	3 270 558	3 307 890
Tout	74 664	6 759 894	6 834 558

TABLE 2.1 – Table de contingence entre le genre et le statut des individus

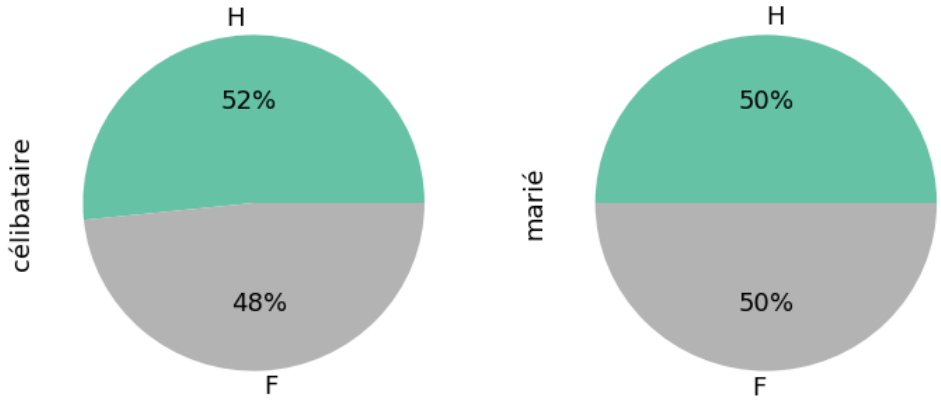


FIGURE 2.1 – Diagramme circulaire des individus par genre

Statut ——————- **Genre**	Marié	Célibataire
Homme	38 527,02	3 488 140,98
Femme	36 136,98	3 271 753,02

TABLE 2.2 – Table de contingence théorique entre le genre et le statut des individus

Celle-ci provient de la ressource [REB2015]. Si la distance est nulle, les deux tableaux sont équivalents et par conséquent, les deux variables sont indépendantes. Par contre, plus la distance est importante et plus les variables sont liées. La distance obtenue pour notre situation vaut $77,37$.

Afin de conclure sur la dépendance ou non, nous avons retourné une p-valeur provenant d'un test d'hypothèses, le test du χ-carré d'indépendance. Premièrement, deux conditions sont à respecter pour pouvoir utiliser ce test : les nombres d'observations au sein des cellules de la table de contingence doivent tous être plus grands que 5 et n doit être supérieur à 20. Si celles-ci ne sont pas d'application, le test du χ-carré n'est pas conseillé car son résultat peut être erroné. Pour notre situation, ces conditions sont respectées. Ensuite, les hypothèses pour ce test d'indépendance sont

- H_0 : les deux variables qualitatives sont indépendantes,
- H_1 : les deux variables sont liées.

C'est évidemment la distance du χ-carré, dont l'expression est donnée à l'équation 2.2, qui est utilisée comme statistique du test. Sous l'hypothèse nulle, cette distance suit approximativement la loi du χ-carré à $(m1-1).(m2-1)$ degrés de liberté, où $m1$ tient pour le nombre de modalités de la première variable et $m2$, de la seconde. Dans notre situation, $m1$, lié au genre, vaut 2 et $m2$, le nombre de modalités du statut, est de 2 puisque l'individu s'est marié en 2000 ou non. Nous avons donc une loi à 1 degré de liberté. Afin d'obtenir la p-valeur associée à ce test, la table de la loi χ-carré est nécessaire et se trouve à l'annexe A.2. Pour trouver une valeur aussi élevée que d^2 observé, avec ddl, le degré de liberté, valant 1, une p-valeur nulle doit être sélectionnée. Le script python retournait 1.42e-18, soit une p-valeur proche de 0. Nous devons donc rejeter l'hypothèse nulle, le genre et la situation des individus sont donc liés. Il faut toutefois être prudent avec ce résultat car nous possédons un jeu de données relativement grand. Les tests d'hypothèses sont plus puissants et donc une petite différence entre deux distributions est considérée comme statistiquement significative. Finalement, nous décidons de conserver cette variable. Bien que les mariages homosexuels ne soient pas légaux et qu'au final, nous aurons autant de femmes que d'hommes mariés, il peut y avoir une différence de nombres dans le

marché du mariage. En effet, certains individus peuvent décider de se marier mais ne trouver personne qui correspond à leurs attentes.

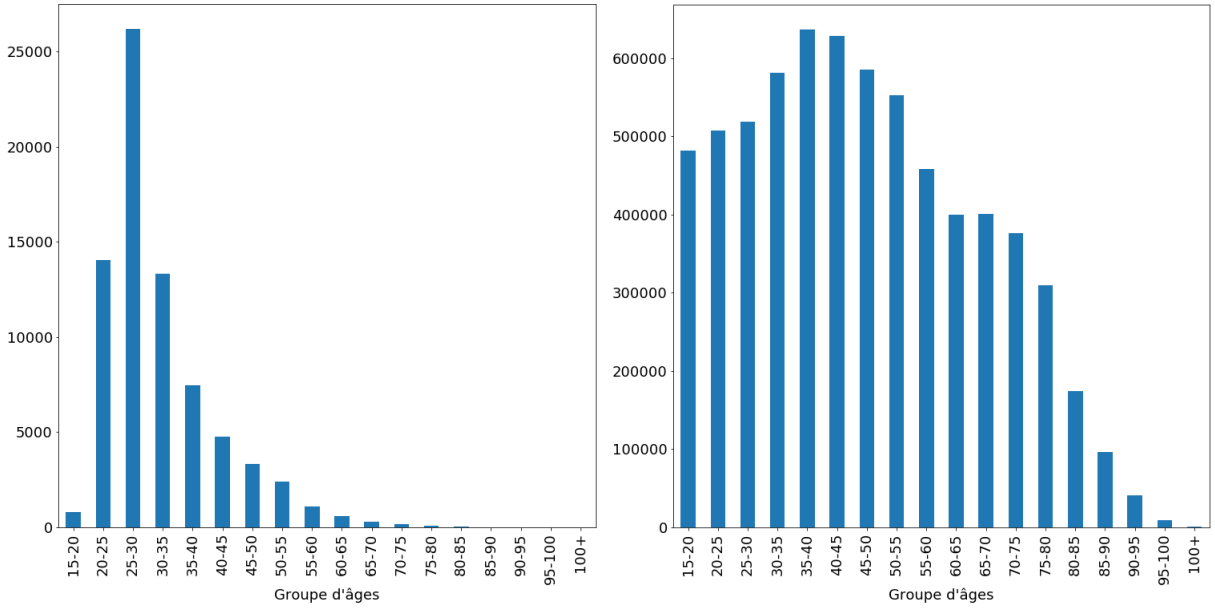
2.1.2 Analyse de l'âge

Similairement à la section précédente, nous avons d'abord construit la table de contingence entre le groupe d'âges des individus et leur situation - si ils ont décidé de se marier en 2000 ou non. Elle est observable à la table 2.3. Cette dernière contenant beaucoup de modalités, il est difficile d'analyser les chiffres obtenus. Nous avons donc réalisé deux graphiques afin de mieux les visualiser, un diagramme circulaire et un histogramme.

Deux histogrammes ont été créés, un pour chaque décision. Ils sont observables à la figure 2.2 et indiquent une forte différence de distributions entre les deux graphiques. L'histogramme (a) affiche un pic de personnes se mariant à l'âge de 25-30 ans et ensuite une diminution qui semble exponentielle. La classe la plus peuplée pour le graphe (b) est celle des 35-40 ans. La diminution du nombre d'individus des catégories plus élevées est moins forte. Ces informations sont aussi observables sur les diagrammes circulaires à la figure 2.3. Toutefois, étant présentées différemment, nous pouvons effectuer une analyse supplémentaire. Notons d'abord que les pourcentages en-dessous de 2% ne sont pas affichés pour une meilleure lisibilité. Ensuite, presque un tiers des personnes s'étant mariées en 2000 avaient entre 25 et 30 ans alors que cette catégorie ne domine pas chez les individus n'ayant pas pris cette décision. De plus, les catégories d'âges des célibataires en 2001 entre 15 et 80 ans sont présentes de manière similaire dans la population. Elles prennent entre 5 et 9% de proportion. Cette étude de graphes indique une différence de distribution d'âges entre les individus s'étant mariés en 2000 et ceux restant seuls. Vérifions maintenant le lien de dépendance entre le groupe d'âges et le statut des personnes.

Statut ——————- **Groupe d'âges**	Marié	Célibataire	**Tout**
15-20	789	481 738	482 527
20-25	14 030	507 470	521 500
25-30	26 204	519 165	545 369
30-35	13 332	581 380	594 712
35-40	7472	636 903	644 375
40-45	4776	628 912	633 688
45-50	3345	585 782	589 127
50-55	2384	552 483	554 867
55-60	1105	457 777	458 882
60-65	607	399 948	400 555
65-70	311	400 360	400 671
70-75	159	376 690	376 849
75-80	101	310 109	310 210
80-85	34	174 227	174 261
85-90	12	96 186	96 198
90-95	3	40 910	40 913
95-100	0	8863	8863
100+	0	991	991
Tout	74 664	6 759 894	6 834 558

TABLE 2.3 – Table de contingence entre le groupe d'âges et le statut des individus



(a) Individus s'étant mariés au cours de l'année 2000

(b) Individus célibataires en 2001

FIGURE 2.2 – Distribution des individus par classe d'âges

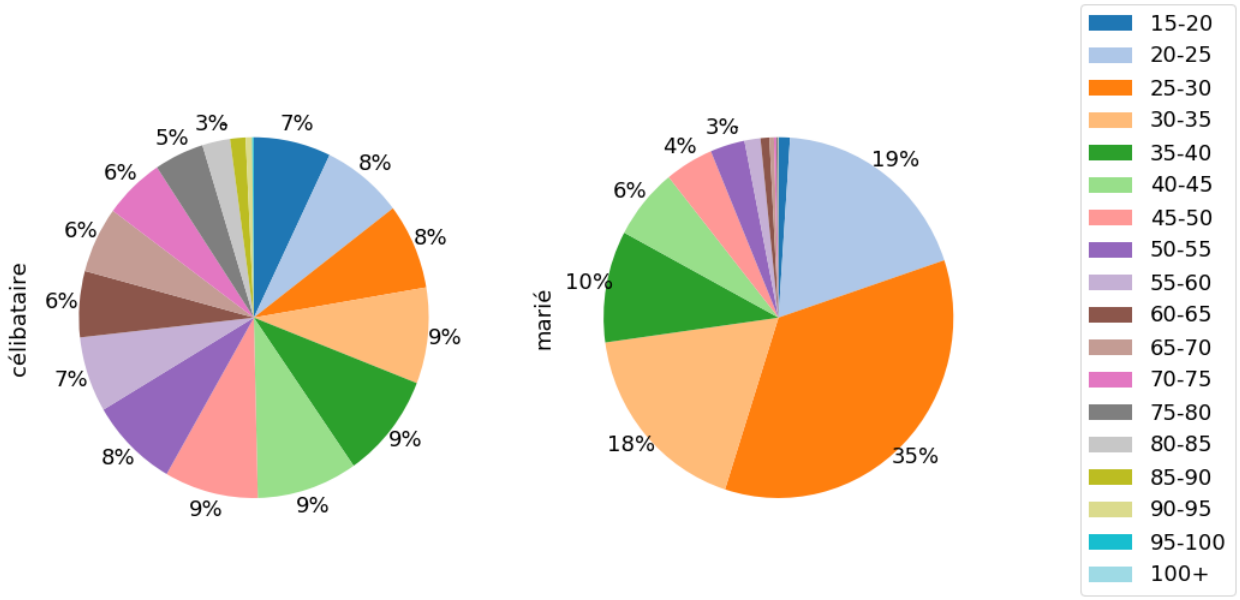


FIGURE 2.3 – Diagramme circulaire des individus par classe d'âges

Nous avons calculé, à la table 2.4, le tableau de contingence théorique de nos deux variables en comparaison. Il diffère de la table calculée précédemment. La distance du χ-carré mesurant cet écart vaut 118 166, 7. La quantité de données traitées peut expliquer cette distance élevée. Il n'est plus nécessaire de vérifier la p-valeur associée au test d'indépendance puisqu'elle est obtenue par le script Python, qui indique aussi les degrés de liberté de la loi χ-carré. Elle est nulle. Cela signifie que les deux variables sont liées. Le groupe d'âges et la décision de se marier ne sont pas indépendants de manière significative.

Statut ——————- **Groupe d'âge**	Marié	Célibataire
15-20	5271,36	477 255,64
20-25	5697,12	515 802,88
25-30	5957,87	539 411,13
30-35	6496,92	588 215,08
35-40	7039,46	637 335,54
40-45	6922,71	626 765,29
45-50	6435,91	582 691,09
50-55	6061,63	548 805,37
55-60	5013,05	453 868,95
60-65	4375,86	396 179,14
65-70	4377,12	396 293,88
70-75	4116,88	372 732,12
75-80	3388,88	306 821,12
80-85	1903,71	172 357,29
85-90	1050,91	95 147,09
90-95	446,95	40 466,05
95-100	96,82	8766,18
100+	10,82	980,17

TABLE 2.4 – Table de contingence théorique entre le groupe d'âges et le statut des individus

2.1.3 Analyse du diplôme

Les deux fichiers utilisés dans cette section contenant des informations sur le diplôme des individus, il nous semblait intéressant de réaliser une courte étude sur cette variable. Nous avons d'abord affiché la table de contingence du diplôme en fonction de la décision de la population. Elle est présente à la table 2.5. Nous observons que beaucoup d'individus s'étant mariés en 2000 ont un diplôme indéterminé. Ensuite, nous retrouvons plusieurs diplômes du secondaire. Ce dernier constat est aussi observable chez les personnes célibataires. La distribution est assez différente entre les deux situations. Pour une meilleure visualisation, nous avons affiché ces informations dans deux diagrammes circulaires, présents à la figure 2.4. L'analyse réalisée sur base de la table de contingence se reflète dans ces graphes. La distribution diffère beaucoup d'un graphe circulaire à l'autre.

Nous terminons par un test d'indépendance du χ-carré afin de conclure sur un lien possible entre le

Statut ——————- **Diplôme**	Marié	Célibataire	**Tout**
IND	52 344	866 028	918 372
PRI	2218	1 024 258	1 026 476
SEC	15 283	3 447 718	3 463 001
SUP	4819	1 421 890	1 426 709
Tout	74 664	6 759 894	6 834 558

TABLE 2.5 – Table de contingence entre le diplôme et la statut des individus

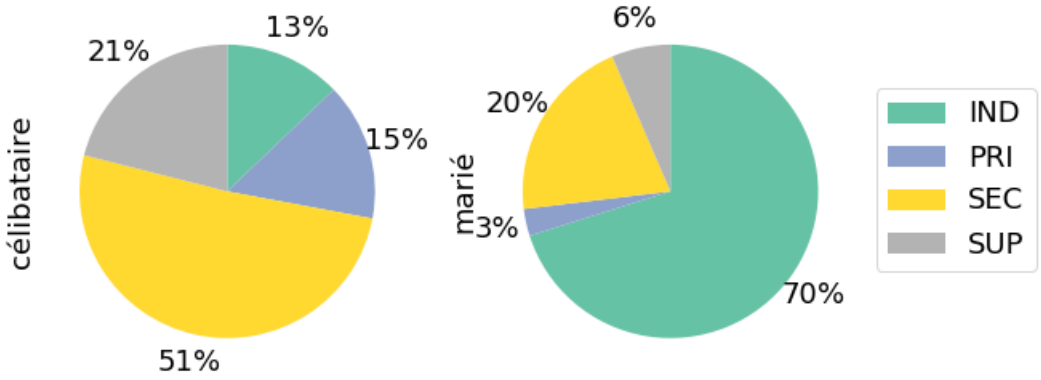


FIGURE 2.4 – Diagramme circulaire des individus par diplôme

diplôme d'un individu et la décision de se marier en 2000. Nous avons d'abord affiché au tableau 2.6 la table de contingence obtenue si nous possédions une situation d'indépendance. Avec une distance du χ-carré de $208\,815,22$, nous pouvons effectivement constater des grandes différences entre la table de contingence obtenue et celle théorique. La p-valeur sortie est nulle indiquant un rejet de l'hypothèse H_0 avec un niveau de certitude élevé. La décision de se marier en 2000 n'est pas indépendante du diplôme des individus.

2.2 Étude de variables influençant le choix de partenaire

Après avoir construit notre marché des mariages avec l'ensemble des individus désireux de se marier, il faut ensuite coupler ceux-ci deux par deux. Notre objectif est d'obtenir des mariages qui soient les plus proches des couples réels. Pour l'atteindre, nous avons tout d'abord analysé les variables à notre disposition en réalisant une étude statistique primaire. Les sections suivantes visent donc à étudier les attributs ayant de l'influence sur le du choix de partenaire dans un ensemble discret.

Nous avons étoffé l'annexe A.1 avec *couples2001.txt* et *suivicouples.txt*, fichiers nécessaires pour les différentes sous-sections subséquentes. Le premier contient peu de variables et concerne les mariages légaux enregistrés en 2001. Il est toutefois important puisqu'il possède des informations sur l'âge du chef de ménage ainsi que celui de son conjoint. Le second fichier concerne aussi les mariages enregistrés à cette date mais comprend d'autres informations comme le diplôme des partenaires. Des explications sur le traitement particulier des données pour ces deux fichiers sont délivrées lors de leur première mention.

Similairement à la section précédente, nous avons affiché à l'annexe A.5 le notebook Jupyter utilisé pour obtenir les graphes et analyses des sections qui suivent. Seuls l'âge et le diplôme des membres

Statut ——————- **Diplôme**	Marié	Célibataire
IND	10 032,74	908 339,26
PRI	11 213,72	1 015 262,28
SEC	37 831,49	3 425 169,51
SUP	15 586,06	1 411 122,94

TABLE 2.6 – Table de contingence théorique entre le diplôme et la statut des individus

du couple sont étudiés. La justification est celle donnée précédemment : peu de variables étaient disponibles. Dans cette section, nous avons de nouveau pris la décision de ne pas analyser la commune ni la santé des individus mariés. Pour leur santé, cela se justifie de manière similaire. Au niveau de leur lieu d'habitation, il est difficile d'étudier l'importance de leur proximité puisque nous ne possédons pas les informations sur l'adresse des individus avant leur mariage. Évidemment, d'autres caractéristiques démographiques entrent en jeu dans le choix du partenaire. Toutefois, les données sont indisponibles et souvent confidentielles. Nous tentons alors, au chapitre 4, de nous approcher au mieux des mariages réels grâce aux variables dont nous disposons.

2.2.1 Analyse de l'âge des partenaires

Cette section débute son étude avec le fichier *couples2001.txt* en affichant une rapide analyse statistique contenant quelques indicateurs importants : la moyenne, l'écart-type, les quartiles, le minimum et maximum. Les résultats sont disponibles à la table 2.7. En observant les informations concernant les genres, nous en déduisons, grâce à la moyenne et aux quartiles, que le chef de ménage est plus souvent de genre masculin. Les résultats obtenus pour sexecm et sexecj sont complémentaires et donc redondants puisqu'en 2001, les mariages homosexuels n'étaient pas légaux. Il existe alors une relation linéaire entre ces deux variables. Le conjoint est toujours de genre opposé à celui du chef de ménage. Ensuite, analysons les informations obtenues pour l'âge des partenaires. Pour le chef de ménage, l'âge se situe entre 19 et 104 ans avec une moyenne dépassant les 50 ans. La médiane est assez proche de cette valeur, signifiant que la moitié des chefs de ménage mariés ont plus de 51 ans. Une même analyse peut être réalisée pour les conjoints puisque les résultats sont assez similaires. Notons tout de même que toutes les valeurs d'indicateurs obtenues pour les chefs de ménage sont plus élevées que celles des conjoints. Enfin, tant pour le chef de ménage que pour le conjoint, l'écart-type est élevé signifiant une grande variation des valeurs autour de la moyenne.

Pour analyser la distribution de l'âge des chefs de ménage et de leur conjoint, deux boîtes à moustaches ont été réalisées. Elles permettent, de surcroit, d'observer les éventuels outliers. Celles-ci sont visibles à la figure 2.5. Ce graphique appuie notre analyse : la distribution de l'âge des conjoints est plus basse que celle des chefs de ménage. Quelques outliers ont été relevés ; ils correspondent à des âges élevés. Ces valeurs sont aberrantes dans le cadre de l'âge du chef de ménage ou du conjoint et non pas dans la globalité. Ce sont des outliers car il existe peu de chefs de ménage ou de conjoints de cet âge-là. Or, le mariage de ces personnes avec leur partenaire n'est peut-être pas aberrant. Nous décidons alors de ne pas utiliser ce graphique pour justifier le retrait d'outliers mais plutôt d'observer le nuage de points à la figure 2.6.

Les coordonnées sur ce graphe représentent les âges des partenaires pour tous les mariages enregistrés jusqu'en 2001. Ayant dans nos données, plusieurs couples de mêmes âges, nous avons ajouté une

	sexecm	sexecj	agecm	agecj
Moyenne	1,02	1,98	52,55	50,25
Ecart-type	0,12	0,12	14,92	14,82
Minimum	1	1	19	16
25%	1	2	40	38
Médiane	1	2	51	49
75%	1	2	64	62
Maximum	2	2	104	101

TABLE 2.7 – Résumé statistique des données du fichier *couples2001.txt*

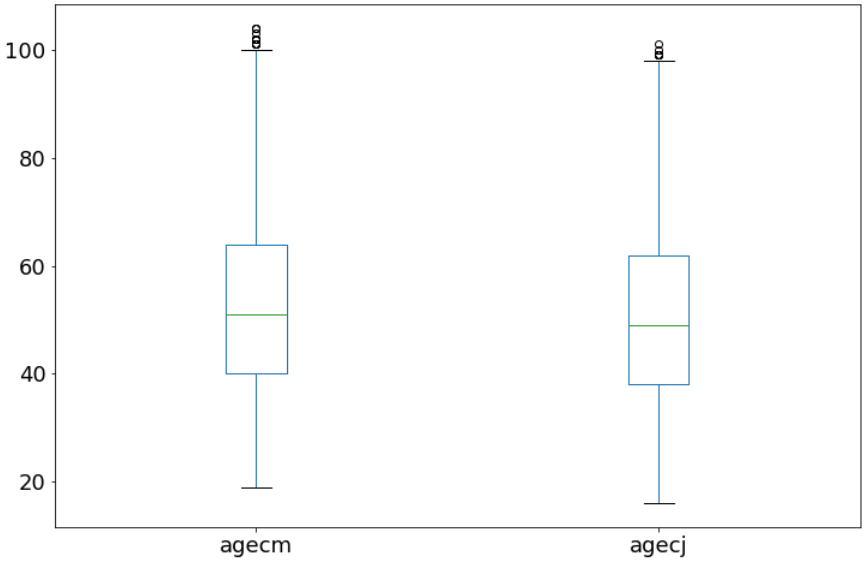


FIGURE 2.5 – Distribution de l'âge des partenaires

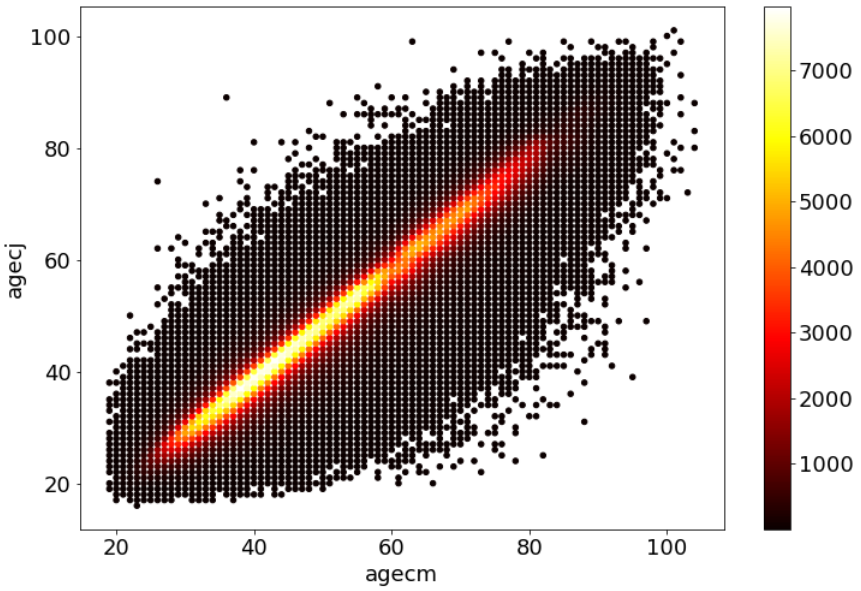


FIGURE 2.6 – Relation entre l'âge des partenaires

dimension supplémentaire sur le graphique : la couleur des points. Cette dernière s'éclaircit avec l'augmentation du nombre de couples de ces coordonnées. Pour les nuages de points qui suivent, nous avons aussi pris en compte cette dimension importante mais pour une meilleure lisibilité, la palette de couleurs est parfois modifiée. Le graphique construit permet donc d'observer les couples d'âges aberrants. Quelques points se détachent de la masse mais nous ne pouvons pas conclure sur leur statut d'outliers. Ce nuage de points indique une autre information importante, la présence d'un lien linéaire positif entre les deux variables. Cela est déductible de la lignée de points de couleur jaune-rouge qui semble former une droite croissante. Pour appuyer cette hypothèse, nous avons d'abord calculé la corrélation de Pearson. Cet indice, noté r, mesure la corrélation linéaire entre deux variables quantitatives, autrement dit

la force d'une relation linéaire entre celles-ci. Il s'obtient par la formule

$$r = \frac{Cov(X,Y)}{\sigma_X \, \sigma_Y},$$

où σ_X représente l'écart-type de la variable X et $Cov(X,Y)$, la covariance entre les variables X et Y. Ses valeurs se situent dans l'intervalle $[-1;1]$ où une valeur positive (négative) indique la présence d'un lien linéaire positif (resp. negatif) entre les variables. Plus r est proche de 1, ou -1, et plus la corrélation est forte. Il faut néanmoins être prudent car même si les deux variables sont fortement corrélées, il n'y a pas nécessairement une relation de causalité entre elles deux. En effet, elles peuvent être liées toutes deux par une causalité commune. La corrélation de Pearson entre agecm et agecj est de $0,96$ indiquant une forte corrélation positive entre les deux variables.

Pour quantifier ce lien entre les variables, nous avons réalisé une régression linéaire. Celle-ci détermine la droite, $Y = aX + b$, qui s'ajuste au mieux à nos données en estimant les paramètres a et b, la pente et le terme indépendant respectivement. La figure 2.7 affiche la droite de régression obtenue, $Y = 0,96X + 0,05$. Sa pente vaut $0,96$ et son terme indépendant $0,05$. Nous retrouvons alors une droite proche de l'identité. Nous avons aussi calculé l'indice du r^2 valant $0,92$ et correspondant au carré du coefficient de corrélation. Il représente la qualité d'ajustement de la droite. Il indique, pour cette situation, que la droite correspond bien aux données.

Avant de conclure cette sous-section, nous avons effectué une étude similaire sur les variables genercm et genercj du fichier *suivicouples.txt*, soit les catégories d'âges des mariés enregistrés jusqu'en 2001. En effet, il arrive souvent que l'âge exact des individus soit absent des données et plutôt exprimé sous forme de classes. Nous voulions en conséquence observer si les résultats obtenus précédemment étaient similaires en groupant les âges par tranche de cinq années.

Pour travailler sur cette variable catégorielle, une modification a dû être encourue afin d'obtenir une variable quantitative et non plus qualitative. Ce changement a pu être réalisé car nous étions en présence de données catégorielles ordonnées. La table 2.8 affiche le codage utilisé pour convertir les variables contenant la classe d'âges du chef de ménage et du conjoint en entiers. Remarquons que les classes

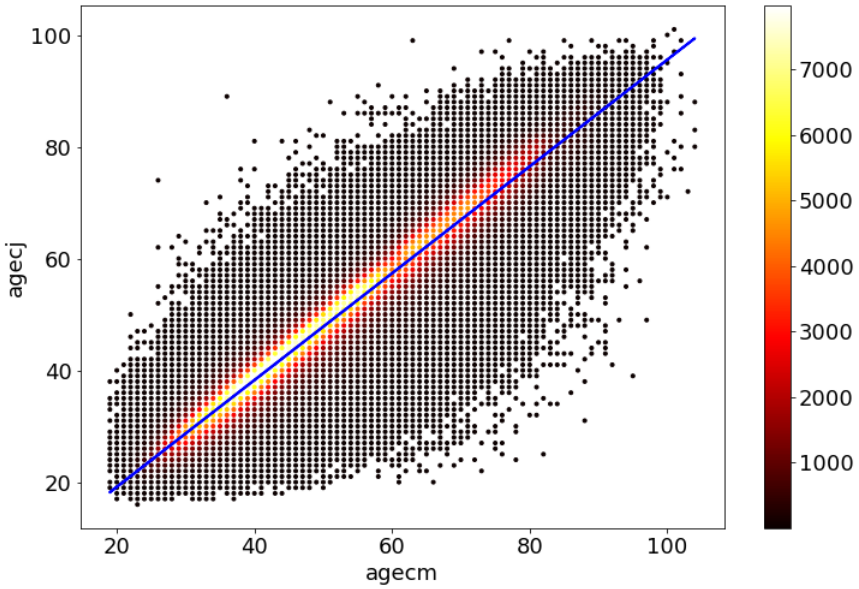


FIGURE 2.7 – Régression linéaire entre l'âge des partenaires

Classe d'âges	15-20	20-25	25-30	30-35	35-40	40-45	45-50	50-55	55-60
Entier correspondant	3	4	5	6	7	8	9	10	11
	60-65	65-70	70-75	75-80	80-85	85-90	90-95	95-100	100+
	12	13	14	15	16	17	18	19	20

TABLE 2.8 – Conversion des classes d'âges en entiers

d'âges de 15 ans ou moins ne sont pas présentes car les individus concernés ne peuvent pas être mariés. Précisons enfin que le groupe quinquennal 15-20, soit le groupe 3, correspond mathématiquement à l'intervalle $[15; 20[$. Ces classes ne se chevauchent donc pas.

Nous avons d'abord affiché les valeurs de quelques mesures importantes en statistiques. Elles sont observables dans le tableau 2.9. Il faut être prudent dans notre interprétation puisque ce sont des variables catégorielles. La moyenne est proche de la médiane, valant 10. Cela signifie que 50% des individus mariés appartiennent à la classe $50 - 55$ ou à une classe supérieure. La médiane, pour le chef de ménage, était précédemment de 51. Cette valeur se situe donc bien dans l'intervalle $[50; 55[$. Par contre, pour le conjoint, elle était de 49. Pour le premier et troisième quartiles, les valeurs correspondent aussi à celles trouvées au tableau 2.7 excepté pour l'âge du chef de ménage pour le premier quartile puisqu'il était de 40. Dans la situation des groupes quinquennaux, le premier quartile appartient à la classe 7, soit $[35; 40[$. Une dernière comparaison se situe au niveau des moyennes. Dans le tableau 2.7, elles sont plus élevées que la médiane alors qu'à présent, elles sont légèrement plus faibles. Cela s'explique par le groupement des âges en cinq ans. Étant donné la similarité entre les distributions des quartiles des classes d'âges du chef de ménage et du conjoint, nous n'avons pas réalisé de boites à moustaches.

Pour étudier les liens entre la catégorie d'âges des individus, un nuage de points est visible à la figure 2.8. Son allure ressemble à celle de la figure 2.6 puisque nous y retrouvons une approximation de droite croissante. De plus, grâce à la couleur des points sur le graphe, nous observons que la majorité des couples se situe le long de cette droite. Il est toutefois intéressant de calculer le coefficient de corrélation pour appuyer cette observation. Celui-ci vaut $0,95$. Il est moins élevé que celui entre agecm et agecj mais reste assez proche. Les conclusions sont donc similaires à celles écrites précédemment, les âges du chef de ménage et de son conjoint sont fortement corrélés positivement. Il est alors important de considérer cette variable lors de la détermination du choix du partenaire pour un individu.

Nous terminons cette section avec la distribution des différences de classe d'âges au sein des couples. L'histogramme la contenant est observable à la figure 2.9. La maxime « Qui se ressemble, s'assemble » est vérifiée puisque la plupart des partenaires possèdent la même catégorie d'âges ou une catégorie de différence. Nous avons chiffré cette analyse dans le tableau 2.10. Il confirme que environ 90% des mariés possèdent des catégories d'âges proches ou identiques. Quelques mariages aux caractéristiques

	genercm	genercj
Moyenne	9,91	9,45
Ecart-type	3,00	2,98
Minimum	3	3
25%	7	7
Médiane	10	10
75%	12	12
Maximum	20	20

TABLE 2.9 – Résumé statistique des données concernant la classe d'âges des personnes mariées

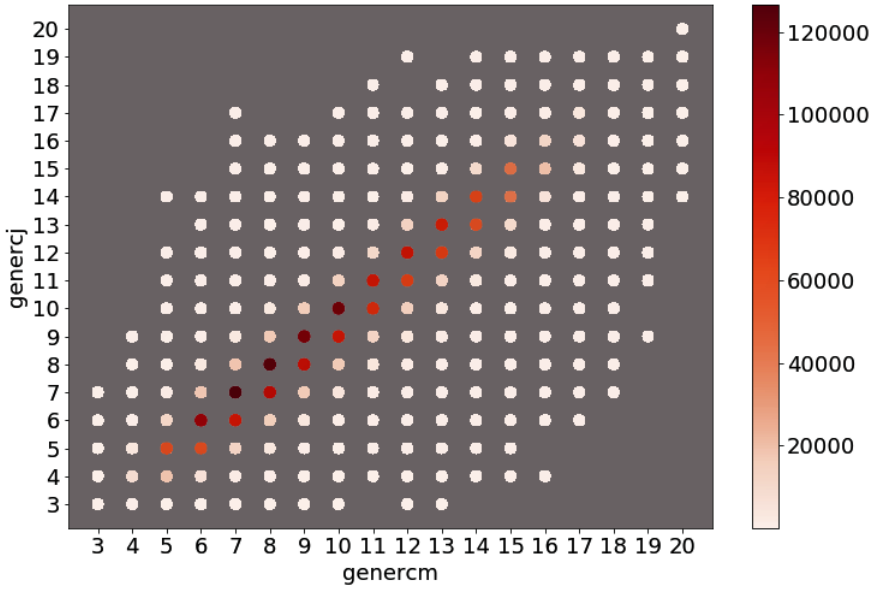


FIGURE 2.8 – Relation entre le groupe d'âges des partenaires

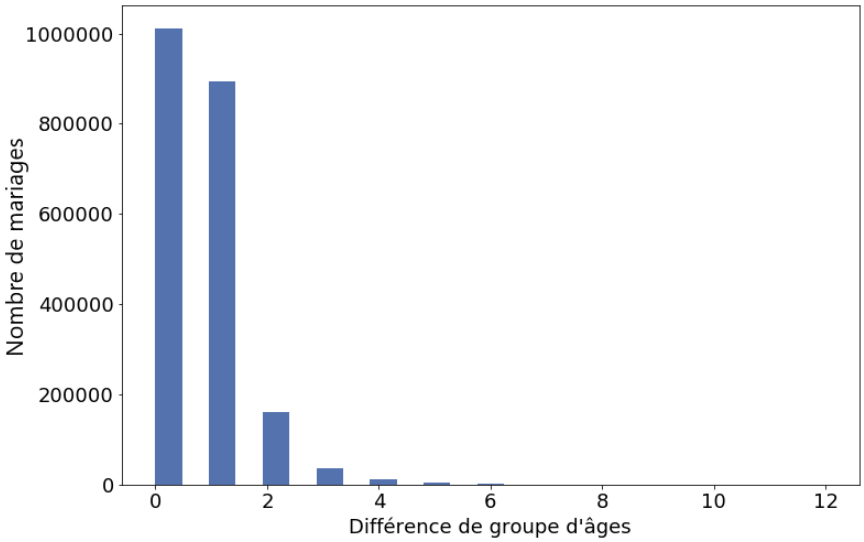


FIGURE 2.9 – Distribution des différences de catégorie d'âges au sein des mariages

extrêmes sont relevés mais ils sont peu nombreux.

2.2.2 Analyse du diplôme des partenaires

Cette section se déroule de manière similaire à la précédente puisque nous sommes, de surcroit, en présence de variables catégorielles ordonnées. Nous avons donc, dans un premier temps, préparé les données en transformant les catégories en entiers afin de permettre le calcul d'indicateurs statistiques. Nous avons affiché la conversion effectuée dans la table 2.11. Nous avons ensuite supprimé les couples où au moins un partenaire avait un niveau de diplôme manquant puisque nous ne pouvions les comparer aux autres mariés. Il nous reste alors $1\,985\,857$ couples dans nos données, représentant environ 93% de la population. Nous avons donc retiré peu de couples. Nous continuons notre analyse puisqu'il reste suffisamment de données pour gagner en compréhension sur les variables dipglcm et dipglcj, définies

Différence de classe d'âges entre les deux conjoints	0	1	2	3	4	5	6
Nombre de couples	1 011 447	893 305	161 681	37 247	11 483	3926	1328
Proportion par rapport à la totalité (%)	47,69	42,12	7,62	1,76	0,54	0,18	0,06
	7	8	9	10	11	12	
	405	137	40	20	4	1	
	0,02	0,01	0	0	0	0	

TABLE 2.10 – Résumé chiffré des différences de catégorie d'âges au sein des mariages

Niveau de diplôme	IND	PRI	SEC	SUP
Entier correspondant	0	1	2	3

TABLE 2.11 – Conversion des niveaux de diplômes en entiers

à l'annexe A.1.

La première étude concerne les indicateurs standards en statistiques. Leurs valeurs sont affichées dans le tableau 2.12. De nouveau, il existe une forte similitude entre les valeurs des indicateurs pour le diplôme des chefs de ménage et celui des conjoints. Une analyse commune est alors réalisée. L'écart type est faible mais cela s'explique par les valeurs prises par les deux variables : un entier entre 0 et 3. Grâce aux quartiles, nous pouvons affirmer que plus de 50% des individus issus de notre population réduite possèdent un diplôme plus élevé que le primaire. De plus, plus de 25% de la population mariée possède un diplôme indéterminé.

Il est ensuite intéressant d'afficher les couples de valeurs dipglcm et dipglcj dans un nuage de points afin d'analyser leur lien. Ce graphique est observable à la figure 2.10. Il indique tout d'abord qu'une majeure partie des partenaires possèdent le même niveau de diplôme, de secondaire ou indéterminé. Ensuite, un lien linéaire est difficilement visible. Nous pouvons toutefois penser que s'il existe une corrélation entre ces deux variables, elle est positive. Vérifions-le avec la valeur du coefficient de corrélation de Pearson. Ce dernier vaut $0,56$. Nous faisons face à une corrélation moyenne positive.

Pour cette étude, nous ne réalisons pas de régression linéaire car nos données, bien que numériques, proviennent d'une transformation de données catégorielles ordonnées. L'interprétation d'une telle visualisation est donc difficile.

Nous terminons cette section dédiée aux diplômes des mariés en analysant la distribution des diffé-

	dipglcm	dipglcj
Moyenne	1,52	1,4
Ecart-type	1,08	1,11
Minimum	0	0
25%	0	0
Médiane	2	2
75%	2	2
Maximum	3	3

TABLE 2.12 – Résumé statistique des données concernant les diplômes des personnes mariées

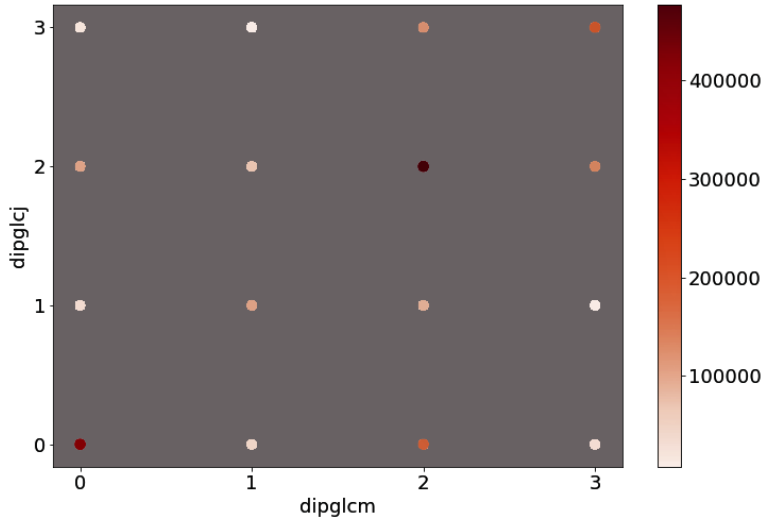


FIGURE 2.10 – Relation entre la classe de diplômes des partenaires

rences de diplômes au sein des couples. La figure 2.11 affiche alors l'histogramme attendu. Une majeure partie des partenaires possèdent des diplômes similaires. Plus la différence s'intensifie et plus nous retrouvons peu de couples. Pour posséder une étude précise de cette situation, nous avons affiché, à la table 2.13, le nombre de couples par différence de diplômes ainsi que la proportion que ce chiffre représente par rapport à la réalité. Plus de la moitié des couples mariés disposent d'un diplôme similaire. De nouveau, l'adage « Qui se ressemble, s'assemble » est d'application.

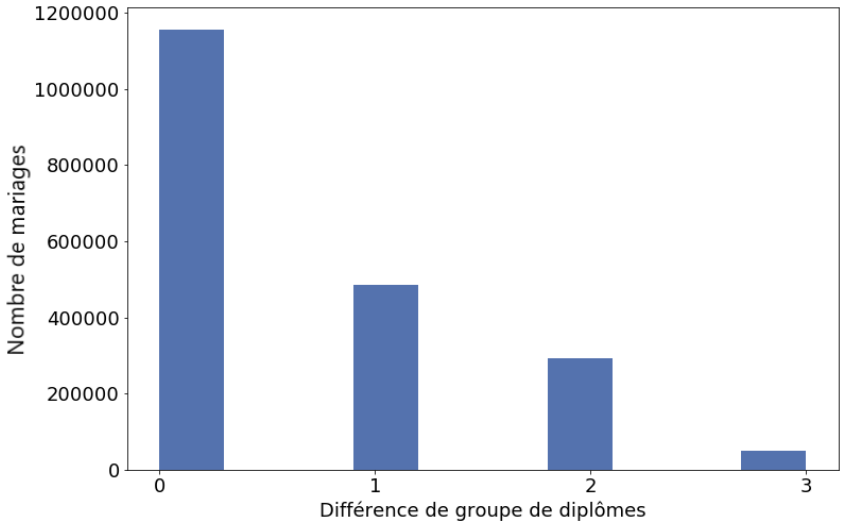


FIGURE 2.11 – Distribution des différences de catégorie de diplômes au sein des mariages

Différence de classe de diplômes entre les deux conjoints	0	1	2	3
Nombre de couples	1 156 215	486 365	292 141	51 136
Proportion par rapport à la totalité (%)	58,22	24,49	14,71	2,58

TABLE 2.13 – Résumé chiffré des différences de catégorie de diplômes au sein des mariages

Chapitre 3

Création du marché du mariage

La première étape lors de la création d'une dynamique des mariages est l'identification des individus qui souhaitent se marier au cours d'une année fixée. Ce chapitre permet alors de créer un marché de mariages, regroupant les agents désireux de se marier. Cette étape reste peu complexe puisqu'elle concerne chaque personne prise individuellement. En effet, dans ce travail, comme dans plusieurs écrits de la littérature, la décision de se marier ne dépend pas des caractéristiques des autres agents dans la population. Plusieurs solutions ont été travaillées afin d'obtenir les meilleurs résultats. Il est important de préciser que nous ne possédons pas d'information sur le désir des individus de s'unir mais bien les données des personnes mariées au cours d'une année particulière. En effet, celles-ci sont compliquées à obtenir. Il faudrait, pour cela, réaliser une enquête spécifique et obtenir assez de réponses pour construire un échantillon significatif de la population étudiée. La conséquence directe de cette remarque est la construction biaisée des marchés qui suivent. Ceux-ci contiennent uniquement la prédiction des individus s'étant effectivement mariés et non pas celle des personnes en ayant le désir.

Tout d'abord, nous utilisons la théorie classique des choix discrets, soit le modèle RUM - Random Utility Maximisation, la maximisation de l'utilité aléatoire - et ensuite une méthode provenant du machine learning, les réseaux de neurones.

Ce chapitre se déroule de la sorte ; nous débutons par l'étude des modèles de choix, avec pour commencer, quelques points théoriques. Ensuite, vient un court texte détaillant le logiciel utilisé, Biogeme. Après cela, une partie concernant l'implémentation du choix discret est écrite. Cette section se finit par la vérification des résultats obtenus. La deuxième partie du chapitre concerne les réseaux de neurones : la théorie, leur implémentation mais aussi leur vérification. Nous terminons cette première partie de solution par une courte conclusion détaillant le choix du modèle le plus approprié.

Sommaire

3.1 Modèle de choix discrets

Depuis sa création, la théorie des choix discrets est un des processus les plus utilisés pour modéliser la demande en transport. En effet, ce domaine est assez riche pour les personnes intéressées par les modèles de choix. Au vu des excellents résultats qu'elle retourne, nous voulons appliquer cette théorie sur notre sujet afin de modéliser la prise de décision des individus dans leur choix de se marier ou non mais aussi dans leur choix de partenaire. C'est sur le premier point que ce chapitre se concentre. Nous débutons cette section avec quelques informations théoriques. Pour les rédiger, nous utilisons principalement les ressources [DUM2017, FEI1998, COR2017].

3.1.1 Théorie

La théorie des choix discrets a pour objectif de modéliser le choix y d'un individu n, extrait d'une population de taille N, parmi un ensemble fini d'alternatives $C_m = \{y_1, \ldots, y_m\}$; cela au moyen de caractéristiques propres à l'individu mais aussi propres aux alternatives. L'ensemble de choix doit satisfaire quelques règles. Les alternatives doivent être mutuellement exclusives et l'ensemble doit être exhaustif. Il n'est pas possible d'énumérer toutes les caractéristiques à prendre en compte dans la décision. Elles sont uniquement connues du décideur. Le choix est donc fonction des attributs connus et inconnus, $y = h(x, \epsilon)$ où

- x représente les caractéristiques connues,
- ϵ représente les caractéristiques inconnues, aussi appelées erreurs.

Le processus de choix discret est par conséquent non-déterministe, le choix de l'agent n'est pas calculé avec certitude. Le modèle implémenté doit alors approcher ce choix au mieux. Pour ce faire, nous calculons les probabilités pour chaque individu de choisir chaque alternative. Soit ϵ aléatoire tel que $\epsilon \sim f(\epsilon)$ et x connu alors

$$P\left(y \mid x\right) = P\left(\epsilon \mid h(x, \epsilon) = y\right).$$

Autrement dit, soit la fonction indicatrice $\mathbb{I}$ telle que

$$\mathbb{I}\left[\, h(x, \epsilon) = y \,\right] = \begin{cases} 1 \text{ si } h(x, \epsilon) = y, \\ 0 \text{ sinon.} \end{cases}$$

Alors

$$\begin{aligned} P(y \mid x) &= P\left(\, \mathbb{I}\left[\, h(x, \epsilon) = y \,\right] = 1 \,\right), \\ &= \int \mathbb{I}\left[\, h(x, \epsilon) = y \,\right] f(\epsilon)\, d\epsilon. \end{aligned}$$

Cette dernière intégrale est déterminée analytiquement ou approchée par simulation.

Le modèle basique est Random Utility Maximisation (RUM), la maximisation de l'utilité aléatoire. Il est toutefois intéressant à utiliser puisqu'il peut modéliser des comportements complexes. La théorie derrière celui-ci émet l'hypothèse que chaque individu choisit l'alternative qui le rend le plus heureux ou, en termes économiques, qui maximise son utilité. Notons, U, la fonction d'utilité qui exprime les préférences des agents. La variable U_{in} représente alors l'utilité perçue par le décideur n pour l'alternative y_i. Ce dernier choisit donc y_i si et seulement si $U_{in} \geq U_{jn}\ \forall j \neq i$. Comme expliqué précédemment, l'utilité réelle, ou utilité aléatoire, n'est jamais connue. Notons alors V, l'utilité calculée par le modèle, qui tente d'approcher U. Nous pouvons réécrire

$$U_{in} = V_{in} + \epsilon_{in}, \tag{3.1}$$

où l'utilité aléatoire est décomposée en deux parties :

- V_{in} est l'utilité connue,
- ϵ_{in} est la partie non observée de U_{in} tel que $\epsilon_n = (\epsilon_{1n}, \ldots, \epsilon_{mn}) \sim f(\epsilon_n)$.

Les différents modèles de choix discrets se spécifient par la définition de $f(\epsilon_n)$.

Détaillons la nécessité de calculer des probabilités pour déterminer le choix de l'agent. En effet, dans la réalité, les individus n'effectuent pas toujours les mêmes choix. Ou encore, avec les mêmes alternatives, les mêmes attributs et les mêmes caractéristiques socio-économiques, deux individus vont peut-être prendre des décisions différentes. De plus, les informations sur le processus décisionnel manquent. Il est donc nécessaire d'introduire une théorie du choix probabiliste. Comme nous supposons que les individus se dirigent toujours vers l'alternative avec l'utilité la plus haute, la probabilité qu'une personne n prenne l'alternative i revient à la probabilité que son utilité soit plus grande que celle des autres. Autrement dit,

$$P_{in} = P\left(U_{in} > U_{jn}, \forall j \neq i\right).$$

Nous pouvons réécrire cette équation de la manière suivante,

$$\begin{aligned} P_{in} &= P\left(V_{in} + \epsilon_{in} > V_{jn} + \epsilon_{jn}, \forall j \neq i\right) \text{ par l'équation 3.1,} \\ &= P\left(\epsilon_{in} - \epsilon_{jn} > V_{jn} - V_{in}, \forall j \neq i\right), \\ &= \int \mathbb{I}\left[\,\epsilon_{in} - \epsilon_{jn} > V_{jn} - V_{in}, \forall j \neq i\,\right] f(\epsilon_n)\, d\epsilon_n. \end{aligned}$$

Il est important de préciser deux informations concernant l'utilité. En effet, seule la différence entre les utilités compte et l'échelle de valeur n'a pas d'importance. Cela entraine qu'une utilité peut être fixée arbitrairement à 0 dans l'ensemble de choix. Par exemple, lors de l'utilisation de modèles de choix discrets pour déterminer si une personne désire se marier ou rester seule, nous fixons l'utilité liée au célibat au nul. Explicitons maintenant l'utilité observable V. Dans un modèle linéaire comme le RUM, elle est définie par la combinaison linéaire des variables pertinentes disponibles dans les données, pondérées par des poids β. Ces poids doivent être calculés pour correspondre au mieux aux données. L'utilité observée d'un individu n pour l'alternative i peut donc s'écrire

$$\begin{aligned} V_{in} &= \beta_1 X_{in1} + \beta_2 X_{in2} + \ldots + \beta_k X_{ink} \text{ où } k \text{ est le nombre d'attributs disponibles,} \\ &= \beta' X_{in}. \end{aligned} \tag{3.2}$$

Dans cette équation, les paramètres $\beta_1, \ldots, \beta_k$ sont égaux pour toutes les alternatives et pour tous les agents. C'est un modèle générique. Le modèle spécifique, quant à lui, possède des paramètres qui diffèrent. Ce dernier s'adapte mieux aux alternatives et aux décideurs mais, en contrepartie, est plus complexe.

Dans ce mémoire, un seul type de modèle RUM est employé, le modèle Multinomial Logit. Ce choix est justifié par sa simplicité mais aussi son bon fonctionnement. Il est, de plus, le modèle le plus utilisé dans la littérature. Ce dernier émet l'hypothèse que les termes d'erreur ϵ_{in} suivent une Loi d'Extrémum Généralisée de type I et sont indépendants et identiquement distribués (iid) pour toutes les alternatives. Ils ont donc même variance et sont non corrélés. Toutefois, ce modèle permet un calcul aisé des probabilités. L'hypothèse d'iid est équivalente à celle d'iia, l'indépendance des alternatives sans importance (independance of irrelevant alternatives). Dans ce modèle, la probabilité P_{in} peut se réécrire sous cette

forme,

$$\begin{aligned} P_{in} &= \frac{\exp(V_{in})}{\sum_{j=1}^{m} \exp(V_{jn})}, \\ &= \frac{\exp(\beta' X_{in})}{\sum_{j=1}^{m} \exp(\beta' X_{jn})} \text{ par l'équation 3.2.} \end{aligned} \tag{3.3}$$

Terminons par parler des critères de détermination de la qualité d'un modèle obtenu. Nous utilisons les deux indices les plus cités dans la littérature : le log-vraisemblance et le ρ^2. Tout d'abord, la fonction de vraisemblance est un critère qui quantifie la probabilité que les individus proviennent effectivement de notre échantillon. Il faut donc le maximiser et cela donne lieu à la méthode dite du maximum de vraisemblance. Soit, P_{in}, la probabilité définie à l'équation 3.3, notons-la à présent $P_{in}(\beta)$ puisqu'elle dépend de ces paramètres. La fonction de vraisemblance, $\mathcal{L}(\beta)$, est définie par le produit des probabilités $P_{in}(\beta)$. Autrement dit,

$$\begin{aligned} \mathcal{L}(\beta) &= \prod_{i=1}^{m} P_{in}(\beta), \\ \Rightarrow \log \mathcal{L}(\beta) &= \sum_{i=1}^{m} \log P_{in}(\beta). \end{aligned}$$

En effet, généralement, c'est le logarithme de la vraisemblance qui est maximisé pour des questions de facilité. Cela n'a pas d'effet sur les valeurs des paramètres. Nous ne pouvons comparer deux modèles au nombre de paramètres différent avec cette valeur de vraisemblance. C'est pourquoi nous utilisons un test du rapport de vraisemblance. Ce dernier rejette l'hypothèse nulle selon laquelle le modèle 1 est le plus adapté si

$$-2\left(\mathcal{L}(\beta_1) - \mathcal{L}(\beta_2)\right) > \chi^2_{1-\alpha,df}$$

où

- $\mathcal{L}()$ est le log-vraisemblance d'un modèle,
- β_p fait référence au vecteur des coefficients de la partie déterministe du modèle p,
- $\chi^2_{1-\alpha,df}$ est le quantile de la loi khi-carré pour le seuil α avec df comme degré de liberté. Nous choisissons α valant $0,05$. Pour calculer df, il suffit d'effectuer la différence entre le nombre de paramètres du second modèle et celui du premier. Le premier modèle est donc celui possédant moins de paramètres.

Enfin, nous pouvons exploiter les valeurs de ρ^2 ajusté obtenues pour comparer les deux modèles. Elles sont comprises entre 0 et 1. Une valeur proche de 1 est signe d'un modèle bien adapté aux données. L'indice ρ^2 ne peut être utilisé pour comparer deux modèles au nombre de paramètres différent, cela justifie donc l'emploi de l'indice ajusté.

3.1.2 Python Biogeme

L'entièreté de cette section repose sur des écrits et publications de M. Bierlaire, auteur du logiciel Biogeme. Nous avons exploité les ressources [BIE2007, BIE2008, BIE2009, BIE2016]. Pour estimer le modèle de choix discrets logit de cette section, nous avons uniquement utilisé le logiciel Biogeme avec sa nouvelle version en Python. Cette section se déroule de la sorte : une brève description du logiciel

est tout d'abord donnée. Ensuite, nous avons détaillé sa nouvelle version dont les spécifications sont écrites en Python. Enfin, les différents algorithmes d'optimisation implémentés au sein de Biogeme sont expliqués.

Biogeme est un logiciel open-source développé par M. Bierlaire en 2003. Son objectif est d'estimer différents modèles basés sur l'utilité aléatoire grâce à la maximisation de la fonction de vraisemblance. Il peut donc étudier les modèles MEV, Multivariate Extreme Value, qui comprennent, notamment, le modèle logit décrit à la section précédente.

La motivation de l'auteur pour construire un tel outil, disponible gratuitement en ligne, vient tout d'abord d'un manque de logiciel approprié. Celui-ci avait besoin d'un outil sophistiqué permettant de résoudre les problèmes modernes fort complexes, notamment dans le domaine du transport. Ensuite, beaucoup d'outils sont destinés à un usage professionnel et sont, par conséquent, payants.

L'auteur destinait son œuvre tant aux chercheurs qu'aux professionnels. L'outil devait, par conséquent, être flexible, extensible et facilement manipulable. De plus, Bierlaire voulait que les utilisateurs ne se préoccupent guère de l'algorithme d'estimation. Il a donc inclus au sein du logiciel plusieurs algorithmes d'optimisation prédéfinis qui offrent, de plus, la possibilité d'explorer différentes pistes.

Biogeme possède d'autres avantages puisqu'il permet aux utilisateurs de spécifier le modèle désiré ainsi que la fonction de vraisemblance associée. De plus, par quelques formulations simples, ils peuvent étudier des modèles plus élaborés.

Le logiciel Biogeme possède deux versions dans des langages de programmation distincts : Bison et Python. La première version de 2003 est écrite en Bison. La seconde, Python Biogeme, assez récente, voit ses spécifications écrites dans le langage Python 3 et son implémentation en C++. La philosophie derrière ces deux versions est assez différente même si les résultats obtenus sont similaires. Nous nous attardons seulement sur la seconde puisque c'est celle utilisée par la suite. Nous avons décidé de travailler uniquement avec cette dernière car selon son auteur, actuellement, il est préférable de se diriger dans cette voie.

Tout d'abord, les entrées spécifiées sont semblables à celles du fichier mod en Bison : la description des paramètres, la spécification de la fonction d'utilité, le choix du modèle, ... Étant donné que des exemples sont fournis sur le site du logiciel, pas ou peu de connaissances en Python sont nécessaires. Toutefois, des utilisateurs maîtrisant ce langage peuvent mettre leurs savoirs à profit en vue de réécrire complètement la spécification du modèle de choix et la fonction de vraisemblance.

Rentrons maintenant au sein du logiciel Biogeme pour en expliquer son mécanisme. La nouvelle version de l'outil se subdivise en deux sous-parties principales :

1. les méthodes et classes C++ pour l'algorithme d'estimation,
2. les modules en Python fournissant des éléments de modélisation ; ceux-ci offrant aux utilisateurs la possibilité de spécifier leur modèle et leur fonction de vraisemblance.

La partie 2 est celle qui est modulable et extensible. De propres modules Python peuvent être ajoutés aux existants par les utilisateurs.

Détaillons maintenant l'estimation de modèles dans Python Biogeme. Pour ce faire, nous avons reproduit un schéma global, figure 3.1. Trois étapes de réalisation sont nécessaires. Tout d'abord, les spécifications du modèle et de sa fonction de vraisemblance associée sont soutirées par l'extracteur

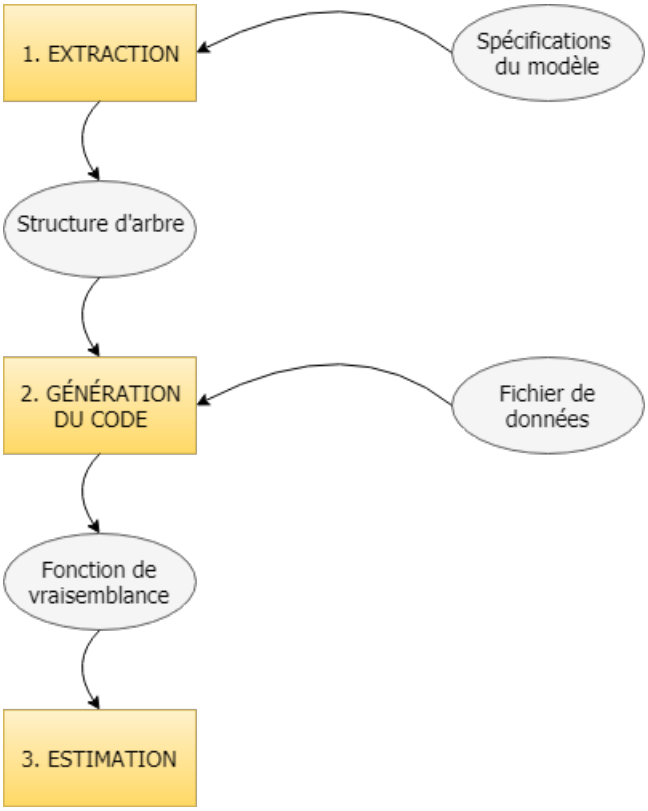


FIGURE 3.1 – Processus interne à Biogeme pour estimer un modèle de choix

Biogeme du fichier Python les contenant. Ensuite, elles sont stockées dans une architecture d'arbre, facilement lisible par le langage C++. De cette structure ainsi que des données entrées, un code en C++ est produit. Les paramètres à estimer et la fonction à optimiser sont alors indiqués au package d'optimisation qui produit enfin les valeurs attendues. Comme précisé précédemment, plusieurs algorithmes d'optimisation sont implantés dans Biogeme. Nous les explicitons ci-après. La motivation liée à l'utilisation du langage C++ pour la partie d'optimisation provient d'un désir d'efficacité.

Différents algorithmes sont donc pré-définis au sein de Biogeme. Ils sont au nombre de cinq : BIO, BIOMC, CFSQP, DONLP2 et SOLVOPT. Avant d'utiliser l'un d'entre eux, il est important de réfléchir quelques instants sur leurs caractéristiques. La première remarque concerne le No Free Lunch théorème énonçant qu'aucun algorithme n'est meilleur que les autres pour n'importe quel modèle. Ils ont donc tous des avantages et des inconvénients. Ensuite, étant donné que ces algorithmes sont construits de différentes manières, il est probable que plusieurs résultats divergents soient obtenus. Toutefois, si il existe quelques différences, elles sont généralement petites. Enfin, aucun des cinq algorithmes n'identifie le maximum global de la fonction de vraisemblance, seulement un maximum local. Une conséquence triviale de cette constatation est qu'un algorithme particulier peut trouver un maximum local différent des autres.

Afin de choisir l'algorithme le plus adapté à un problème particulier, plusieurs règles peuvent être suivies. Si il n'existe pas de contrainte non-triviale sur les paramètres, BIO peut être utilisé. Sinon, ou si cet algorithme est trop lent, CFSQP peut être un bon choix. Il faut être vigilant car ce dernier n'était pas inclus dans la version 1.6 de Biogeme et était payant. Certains utilisateurs n'y ont donc pas accès. Une autre alternative est de prendre DONLP2 qui est moins rapide que CFSQP, puis SOLVOPT. Ce dernier peut parfois être très lent car il n'a pas été construit dans le but de résoudre des types de

problèmes d'optimisation présents dans Biogeme. Si aucun algorithme ne donne des résultats, il faut alors redéfinir le modèle. Ces conseils proviennent de la ressource [BIE2009].

Deux algorithmes d'optimisation sont exploités dans notre mémoire BIO et CFSQP. Ce choix est motivé à la section suivante. Nous allons alors les définir brièvement tous les deux.

- BIO (Bierlaire's Optimization) est un algorithme spécialement adapté pour le logiciel Biogeme. Il repose sur une méthode de gradient conjugué tronqué pour résoudre un sous-problème de région de confiance. Les problèmes qu'il traite doivent posséder de simples limites et pas de contrainte non-triviale.
- CFSQP est, tout d'abord, un algorithme payant lors de la version 1.6. de Biogeme. C'est l'implémentation en C++ de la méthode d'optimisation FSQP (Feasible Sequential Quadratic Programming). Celui-ci est un algorithme newtonien à appliquer aux conditions d'optimalité du premier ordre pour le problème considéré.

Utilisation du logiciel

Avant de détailler l'implémentation de nos modèles de choix, nous trouvons important de décrire brièvement l'utilisation de Biogeme et notamment la démarche pour obtenir les résultats attendus, soit la définition des fonctions d'utilité.

Le jeu de données considéré doit, tout d'abord, respecter certaines conditions avant d'être injecté dans Biogeme. Il doit être unique et par conséquent, contenir toutes les informations nécessaires sur la population. La première ligne du fichier doit être réservée pour la liste des variables à considérer. Chaque ligne suivante concerne une seule observation et possède le même nombre d'éléments que de variables. Le jeu de données n'est pas autorisé à contenir des valeurs nulles.

Ensuite, pour obtenir le modèle estimé par Biogeme, plusieurs simulations sont souvent nécessaires. Lors de la première utilisation du logiciel, il suffit de rentrer en console, positionnée dans le répertoire contenant les documents, la commande *Pythonbiogeme model data.txt* où *model* est le nom du modèle, suivi du nom du fichier comprenant les données. Après son exécution, nous obtenons un fichier de sortie affichant, entre autres, la valeur des différents paramètres. Ce fichier comprend aussi les valeurs de plusieurs indicateurs d'adéquation du modèle, dont les principaux sont le log-vraisemblance et le ρ^2. Un diagnostic est aussi apporté, signalant toute erreur lors de l'exécution.

De plus, au sein du fichier de sortie reçu, une colonne nous indique si il est pertinent ou non de conserver certains coefficients de paramètres dans notre modèle. Pour les identifier, un test d'hypothèses est réalisé dans Biogeme qui retourne la p-valeur associée. Ce test statistique est un t-test robuste avec un seuil de certitude que nous fixons à 95%. L'hypothèse nulle, H0, annonce qu'il ne faut pas considérer le paramètre en question et son coefficient peut alors être fixé à 0 tandis que H1 affirme qu'il doit être pris en compte. Autrement dit, si une p-valeur supérieure au seuil $0,05$ est obtenue pour un paramètre, nous en concluons que ce dernier n'a pas d'influence sur la décision des individus. Inversement, si la p-valeur est plus petite que ce nombre, le coefficient du paramètre sera significativement différent du 0.

Les paramètres n'ayant pas d'influence sur notre modèle sont reconnaissables par un signe distinctif, une étoile. Il est important de les enlever un à un et de relancer une simulation entre chaque retrait puisque les valeurs du t-test robuste varient. De plus, précisons que l'ordre dans lequel nous figeons ces paramètres a une influence sur le modèle final. Nous décidons alors de fixer à chaque simulation, le paramètre ayant la plus haute p-valeur. Enfin, précisons que le test d'hypothèses est influencé par les

bornes choisies pour ces différents paramètres. Il faut, par conséquent, veiller à les prendre suffisamment grandes ou à les adapter afin de ne pas avoir de soucis dans les résultats finaux.

Quand plus aucune p-valeur ne dépasse le seuil de $0,05$, tous les paramètres restants ont une influence significative sur notre modèle de choix. Le modèle final est donc construit grâce aux valeurs des coefficients de paramètres sorties par Biogeme.

3.1.3 Implémentation des modèles

Avant de commencer l'implémentation de notre choix discret, une étape de préparation des données est nécessaire. Le notebook Python utilisé dans cet objectif est présent à l'annexe A.6. Nous avons principalement assemblé les individus célibataires et ceux qui se sont mariés en 2000 dans un jeu de données commun. Une étape essentielle dans la création de modèles prédictifs est la séparation des données en ensemble d'entrainement et ensemble de test. Cela permet, une fois le modèle de choix discrets calibré sur les données d'entrainement, de le tester avec un autre ensemble indépendant du premier. En effet, il est possible que le modèle obtenu soit biaisé, qu'il ait des performances élevées sur les données d'entrainement mais faibles sur de nouvelles données. Or, notre objectif est de posséder un modèle général, qui puisse être utilisé avec des données supplémentaires - par exemple, les célibataires de l'année suivante. Nous avons choisi de diviser notre ensemble d'individus en 70 et 30% pour l'ensemble d'entrainement et celui de test respectivement. Une dernière remarque concerne les cohabitants légaux. Nous avons mentionné précédemment, l'importance de les distinguer des autres célibataires de la population. En effet, ils ne doivent pas être couplés à un individu du marché mais bien à leur cohabitant puisqu'ils prennent la décision de changer de statut civil. Néanmoins, les données reçues ne distinguent pas ces personnes des célibataires au sein de la population. Nous n'effectuons donc pas de traitement spécial.

Nous pouvons, à présent, concevoir les spécifications du modèle de choix en Python. Le script écrit dans ce langage et injecté dans Biogeme est observable à l'annexe A.7. Nous avons commencé nos modèles en traitant l'algorithme BIO puisque c'est le premier proposé dans la littérature. Certains packages Python sont nécessaires. Toutefois, leur installation est inutile puisqu'ils sont inclus dans le téléchargement du logiciel Biogeme.

Il faut ensuite définir les paramètres qui doivent être estimés. Notre modèle possède deux alternatives, rester célibataire ou décider de se marier au cours d'une année particulière, soit 2000 dans ce chapitre. Les informations des individus prises en compte sont le genre, la catégorie d'âges ainsi que de diplômes. Nous avons d'abord décrit les constantes d'alternatives spécifiques, visibles à la figure 3.2. Celle concernant le choix de rester célibataire est fixée à 0. En effet, comme explicité précédemment, seule la différence entre les utilités compte. Nous avons ensuite caractérisé les coefficients des variables, à la figure 3.3. Nous avons délibérément décidé de définir des variables binaires pour les catégories d'âges puisque nous pensons que la relation entre le choix des individus et leur âge n'est pas strictement croissante (ou décroissante). Cette hypothèse est soutenue sur la figure 2.2 (a). La section suivante confirme notre réflexion. Suite à cette décision, nous avons défini de nouvelles variables au sein du script analysé actuellement. Le coefficient lié à la catégorie d'âges 20, soit les 100 ans et plus, a été fixé au nul. Cela permet d'éviter la redondance. En effet, si une personne n'appartient à aucune

```
ASC_married = Beta('ASC_married', 0, -100, 100, 0, 'Married cst')
ASC_single = Beta('ASC_single', 0, -100, 100, 1 'Single cst')
```

FIGURE 3.2 – Constantes spécifiques aux alternatives du premier modèle de choix

```
B_sexe = Beta('B_sexe', 0, -100, 100, 0, 'Sexe')
B_diploma = Beta('B_diploma', 0, -100, 100, 0, 'Diploma')

B_age_G3 = Beta('B_age_G3', 0, -100, 100, 0, 'Age cat. 3')
B_age_G4 = Beta('B_age_G4', 0, -100, 100, 0, 'Age cat. 4')
B_age_G5 = Beta('B_age_G5', 0, -100, 100, 0, 'Age cat. 5')
B_age_G6 = Beta('B_age_G6', 0, -100, 100, 0, 'Age cat. 6')
B_age_G7 = Beta('B_age_G7', 0, -100, 100, 0, 'Age cat. 7')
B_age_G8 = Beta('B_age_G8', 0, -100, 100, 0, 'Age cat. 8')
B_age_G9 = Beta('B_age_G9', 0, -100, 100, 0, 'Age cat. 9')
B_age_G10 = Beta('B_age_G10', 0, -100, 100, 0, 'Age cat. 10')
B_age_G11 = Beta('B_age_G11', 0, -100, 100, 0, 'Age cat. 11')
B_age_G12 = Beta('B_age_G12', 0, -100, 100, 0, 'Age cat. 12')
B_age_G13 = Beta('B_age_G13', 0, -100, 100, 0, 'Age cat. 13')
B_age_G14 = Beta('B_age_G14', 0, -100, 100, 0, 'Age cat. 14')
B_age_G15 = Beta('B_age_G15', 0, -100, 100, 0, 'Age cat. 15')
B_age_G16 = Beta('B_age_G16', 0, -100, 100, 0, 'Age cat. 16')
B_age_G17 = Beta('B_age_G17', 0, -100, 100, 0, 'Age cat. 17')
B_age_G18 = Beta('B_age_G18', 0, -100, 100, 0, 'Age cat. 18')
B_age_G19 = Beta('B_age_G19', 0, -100, 100, 0, 'Age cat. 19')
B_age_G20 = Beta('B_age_G10', 0, -100, 100, 1, 'Age cat. 20')
```

FIGURE 3.3 – Coefficients des variables prises en compte pour le premier modèle de choix

catégorie d'âges entre 3 et 19, elle possède obligatoirement un âge supérieur à 100 ans.

Après ces spécifications, il faut définir les fonctions d'utilité. Celles-ci sont visibles à la figure 3.4. L'utilité v2, celle du choix de rester célibataire, est nulle car la constante spécifique vaut 0. Cela s'explique comme suit : seule la différence entre les utilités compte et leur échelle de valeur n'a pas d'importance. Une utilité peut donc être fixée à 0. Il suffit alors de calculer la probabilité de se marier pour trouver celle de rester célibataire.

D'autres paramètres doivent encore être explicités : le choix du modèle - ici un logit -, la fonction de vraisemblance ou encore l'algorithme d'optimisation. Ce dernier est, pour le premier modèle, BIO dont nous discutons le modèle final dans la section suivante.

3.1.4 Vérification

Dans le cadre de notre mémoire, non pas un mais quatre modèles de choix discrets ont été implémentés. Nous n'affichons pas leurs spécifications puisqu'elles sont similaires à celles présentes à l'annexe A.7 avec quelques simples modifications. Nous voulions approfondir plusieurs pistes pour obtenir des résultats satisfaisants. Nous avons, en conséquence, implémenté un modèle en gardant cet algorithme d'optimisation mais en enlevant les données concernant le diplôme des individus. Ce choix est justifié par l'analyse réalisée à la section 2.1 qui indique une forte présence de diplômes indéterminés dans

```
v1 = ASC_married * one + B_sexe * sexe + B_diploma * diploma
+ B_age_G3 * GAge3 + B_age_G4 * GAge4 + B_age_G5 * GAge5
+ B_age_G6 * GAge6 + B_age_G7 * GAge7 + B_age_G8 * GAge8
+ B_age_G9 * GAge9 + B_age_G10 * GAge10 + B_age_G11 * GAge11
+ B_age_G12 * GAge12 + B_age_G13 * GAge13 + B_age_G14 * GAge14
+ B_age_G15 * GAge15 + B_age_G16 * GAge16 + B_age_G17 * GAge17
+ B_age_G18 * GAge18 + B_age_G19 * GAge19 + B_age_G20 * GAge20

v2 = ASC_single * one
```

FIGURE 3.4 – Utilités liées au premier modèle de choix

la population mariée. Nous avons aussi choisi d'explorer un autre algorithme d'optimisation, CFSQP dont nous disposons gratuitement. C'est un avantage d'étudier plusieurs voies pour différentes raisons mentionnées à la section 3.1.2. Au niveau de cet algorithme, nous avons également implémenté un modèle contenant les informations sur le diplôme des agents et un autre n'en tenant pas compte.

Pour une analyse détaillée de notre méthode de vérification du résultat des choix discrets, un notebook Jupyter est disponible à l'annexe A.8. Nous utilisons l'équation 3.3, se reposant sur les utilités, pour obtenir les probabilités de choix de chaque individu dans le jeu de données. Vu la présence importante d'aléatoire au sein de notre code, nous avons décidé de réaliser plusieurs simulations pour éviter une situation extrême. À chaque itération, nous calculons le total de personnes se mariant ainsi que la prédiction du choix de chaque individu. Toutefois, nous conservons uniquement ces sélections pour la dernière simulation. En effet, nous pouvions calculer des moyennes de ces prédictions mais cela ne nous semblait pas judicieux d'obtenir un choix différent de 0 ou de 1.

Pour la vérification de notre modèle, nous travaillons avec les données de test qui sont prévues à cet effet. Nous avons arbitrairement choisi d'effectuer dix simulations. C'est un bon compromis, selon nous, entre un temps d'exécution pas trop long et un nombre assez important pour éviter les itérations extrêmes. Discutons de ces deux caractéristiques en commençant par la notion temporelle. Tout d'abord, la machine sur laquelle l'entièreté des codes tourne est un Lenovo au processeur Intel(R) Core(TM) i7-2620M, avec une fréquence de $2,70$ GHz, 2 cœurs et 8 Go de mémoire RAM. Les dix simulations se sont exécutées en $12\,203$ secondes, soit 3 heures 23 minutes et 23 secondes. Nous avons affiché l'évolution de ce temps, par itération à la figure 3.5. Ce graphique indique une progression presque linéaire du temps d'exécution. Cela signifie que les itérations s'exécutent sur une période temporelle similaire. Notons qu'une itération demande en moyenne 1220 secondes ou 20 minutes et 20 secondes.

Pour analyser les fluctuations des prédictions entre les différentes simulations, nous avons affiché, à la figure 3.6, la prévision du nombre de personnes au sein du marché des mariages par itération. Ces nombres oscillent mais restent dans une étroite rangée de valeurs : entre $22\,100$ et $22\,700$. Étant donné que seulement dix simulations ont été réalisées, ces bornes pourraient être agrandies. Néanmoins nous

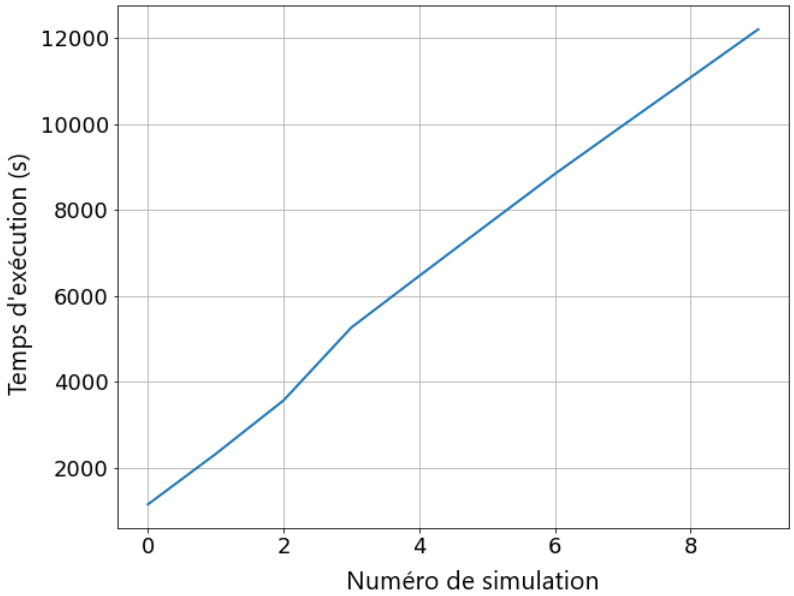


FIGURE 3.5 – Progression du temps d'exécution du premier modèle de choix avec les données de test en fonction des itérations

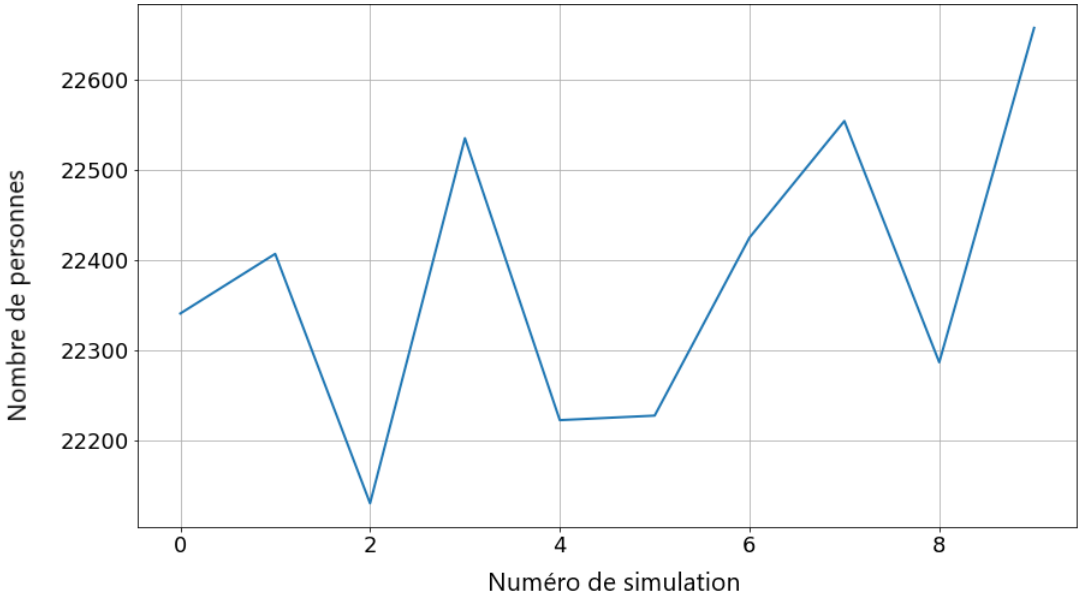


FIGURE 3.6 – Prédiction du nombre de personnes au sein du marché des mariages par simulation pour le premier modèle de choix avec les données de test

ne notons pas de variations importantes dans le nombre de personnes désirant se marier. Nous n'effectuons donc pas de simulation supplémentaire.

Avant d'analyser les résultats de la dixième itération, nous avons affiché, à la figure 3.7, les nombres de personnes désirant se marier en 2000 tant en réalité que prédits. La notation « personne désirant se marier » est trop optimiste. En effet, comme expliqué au début de ce chapitre, nos données ne contiennent pas d'information sur l'envie des individus de se marier une année particulière. Nous considérons alors que toutes les personnes célibataires dans nos données n'avaient aucune aspiration au mariage en 2000. En réalité, nos prédictions tentent de reproduire le nombre de personnes s'étant effectivement mariées cette année-là. Néanmoins, faute de donnée supplémentaire, nous considérons que suite à l'exécution du modèle de choix, le marché formé contient des individus désirant se marier. Ils sont alors au stade de la volonté de trouver un partenaire mais peuvent rester célibataires si leur recherche se solde par un échec. Les couplages sont effectués dans le chapitre suivant. Le graphique 3.7 affiche donc le nombre réel de personnes s'étant mariées en 2000 et ceux prédits par le modèle de choix discrets. Nous observons que les prédictions oscillent autour du nombre réel. Peu de différences sont donc constatées. Nous avons 22 402 mariés en 2000 dans nos données de test et une moyenne de prédiction de 22 379, soit 23 personnes de différence. Remarquons que la dernière simulation, celle dont nous gardons le choix des individus, est celle dont le nombre de personnes est le plus élevé. Nous ne pouvons néanmoins l'écarter puisque travaillant avec de l'aléatoire, nous risquons de rencontrer cette situation à plusieurs reprises. Nous continuons alors cette analyse avec un marché des mariages contenant 22 657 célibataires. Il faut maintenant vérifier si cette prédiction possède une distribution d'individus semblable à la réalité.

Pour approfondir cette analyse, nous avons réalisé une matrice de confusion ainsi qu'un diagramme en bâtons. Discutons, tout d'abord, de cette première sorte de visualisations. Les matrices de confusion permettent de calculer la précision d'un modèle de classification puisqu'elles contiennent des informations sur les données actuelles et celles prédites. Les choix discrets peuvent être considérés comme des algorithmes de classification puisqu'ils permettent de déterminer la décision d'un individu parmi un ensemble discret de choix ; soit, de manière équivalente, à classer les individus dans un ensemble discret de catégories. Pour la suite des explications, nous nous aidons d'une matrice de confusion théo-

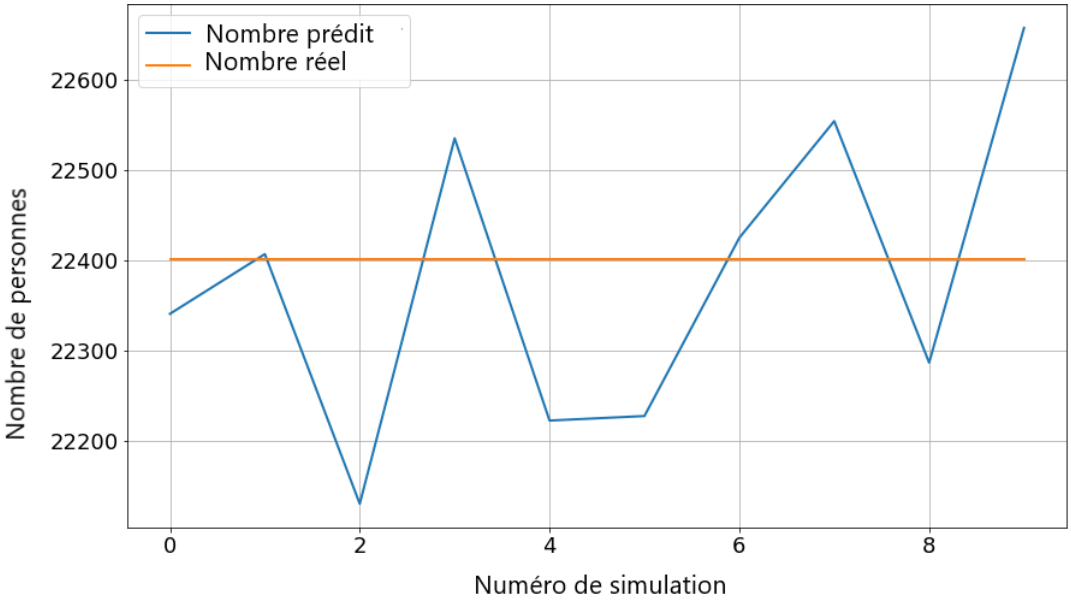


FIGURE 3.7 – Nombre de personnes désirant se marier par simulation pour le premier modèle de choix avec les données de test

rique, présente à la figure 3.8. Détaillons le calcul de l'accuracy, la précision du modèle, en nous aidant de ce schéma. Nous avons représenté une situation théorique comprenant deux classes : Oui et Non. Explicitons les notations utilisées. Les individus de la classe Oui étant prédits comme des Oui sont des vrais positifs (TP), ceux de ce même groupe mais classés en Non, sont des faux positifs (FP). Ensuite, dans le groupe Non, les individus prédits avec ce label sont des vrais négatifs (TN) et ceux classés en Oui par l'algorithme sont des faux négatifs (FN). Pour déterminer l'accuracy, il suffit d'utiliser la formule suivante,

$$\frac{TP + TN}{TP + FN + FP + TP}.$$

La matrice de confusion du premier modèle de choix, présente à la figure 3.9, permet d'annoncer que le modèle n'est pas mauvais dans le sens où il a prédit $2\,008\,683 + 3374 = 2\,012\,057$ labels corrects sur $2\,050\,368$, soit une accuracy de $98,13\%$. Nous remarquons que les classes sont de taille différente : seulement $19\,028 + 3374 = 22\,402$ personnes se sont mariées sur $2\,050\,368$ individus. Cela représente

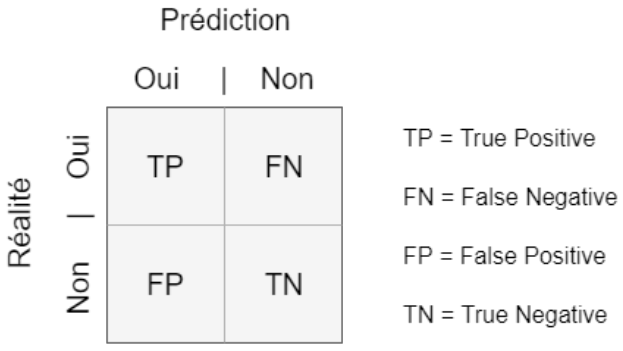


FIGURE 3.8 – Matrice de confusion théorique

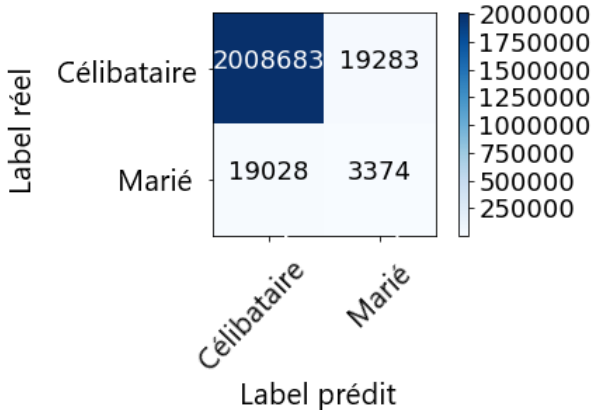


FIGURE 3.9 – Matrice de confusion pour le premier modèle de choix avec les données de test

environ 1% de notre population totale. Cette matrice nous indique que $2\,008\,683$ célibataires en 2001 ont été prédits avec un même statut par notre modèle et que 3374 personnes mariées se retrouvent dans notre marché. Nous observons aussi que $19\,283$ personnes célibataires ont été prédites, par notre choix discret, comme désirant se marier et 19028 individus s'étant mariés en 2000 sont prévus célibataires.

Nous avons utilisé une autre métrique pour mesurer la qualité du modèle en normalisant cette matrice de confusion. Le résultat est observable à la figure 3.10. Cette matrice annonce que la prédiction de TP est de 99%. En revanche, nos résultats contiennent 85% de mauvaises prédictions au niveau des personnes mariées. La plupart de celles-ci sont classées dans le groupe des célibataires par notre modèle. Elles ne font donc pas partie du marché des mariages.

Finalement, pour étudier ces mauvaises classifications, nous avons engendré deux graphiques en bâtonnets, un pour les hommes, le second pour les femmes. Nous décidons de réaliser cette analyse en séparant les personnes de genre opposé car nous pensons que les caractéristiques démographiques des personnes désirant se marier diffèrent entre les hommes et leurs homologues féminins. Ces graphes affichent le nombre de personnes se mariant effectivement en 2000 et les prédictions au sein du marché par catégorie d'âges et sont visibles à la figure 3.11. Nous remarquons que les deux distributions ne sont pas similaires, tant pour les hommes que pour les femmes. Nous avons quantifié ces différences en additionnant, par catégorie d'âges, la valeur absolue des écarts observés. Pour les hommes, nous obtenons 1474 personnes de différence et pour les femmes, 1647. Néanmoins l'allure des distributions

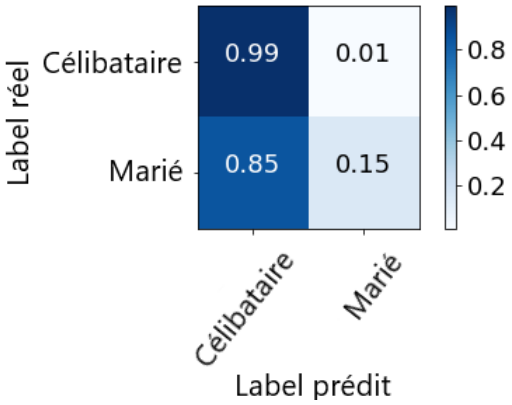
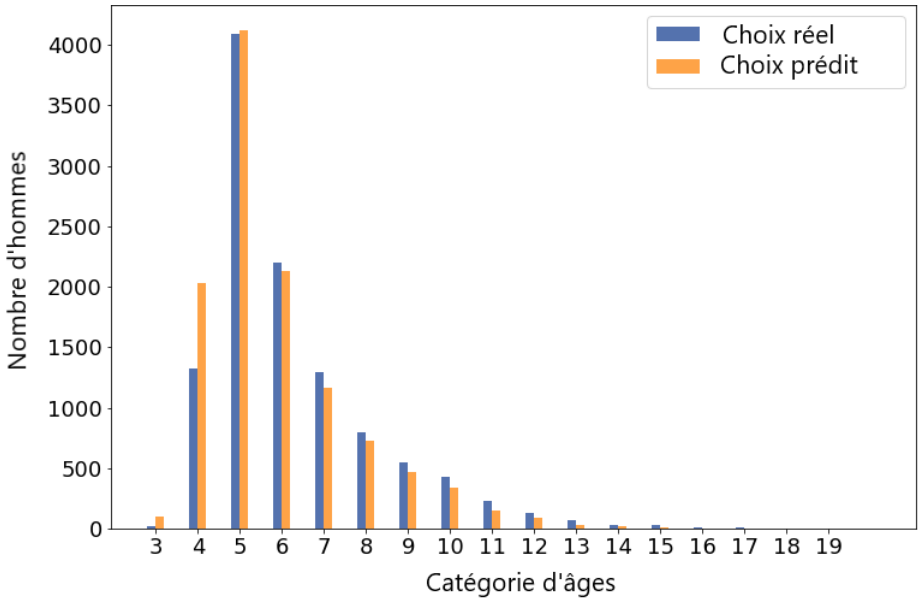
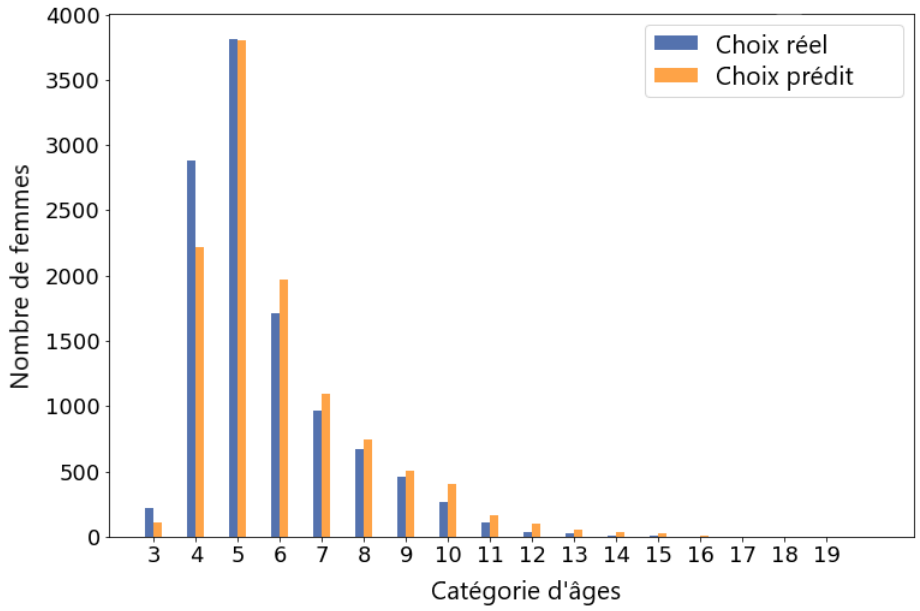


FIGURE 3.10 – Matrice de confusion normalisée pour le premier modèle de choix avec les données de test



(a) Résultat concernant les hommes



(b) Résultat concernant les femmes

FIGURE 3.11 – Nombre de personnes au sein du marché des mariages par catégorie d'âges pour le premier modèle de choix avec les données de test

semble analogue. Nous retrouvons, tant en réalité que dans les prédictions pour les individus masculins et féminins, une croissance du nombre de personnes voulant se marier en 2000 de la troisième à la cinquième catégorie puis une baisse exponentielle avec peu de personnes âgées. Nous observons enfin que la distribution réelle diffère d'un genre à l'autre ; les femmes ont tendance à se marier plus jeunes que les hommes.

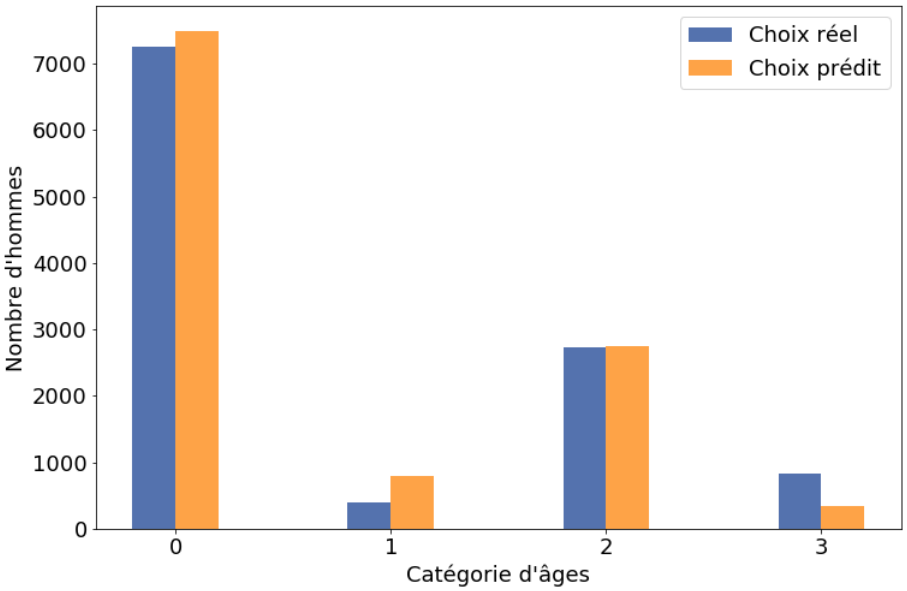
La deuxième caractéristique démographique dont nous connaissons les informations concerne le diplôme des individus. Nous avons réalisé l'étude précédente en considérant, à présent, les classes de diplômes et non plus les catégories d'âges. Les deux graphes en bâtonnets sont disponibles à la figure 3.12 pour les hommes et les femmes. Bien que les distributions des deux graphes semblent similaires, quelques différences sont remarquables, spécialement au niveau de la visualisation (b). Nous avons quantifié ces divergences par catégorie et les avons sommées. Nous obtenons 1154 individus de différence entre le nombre d'hommes et la prédiction du modèle de choix. Pour les femmes, cet écart vaut 2547.

Bien que notre premier modèle de choix prédit un nombre cohérent d'individus au sein du marché des mariages, vis-à-vis de nos données réelles, les caractéristiques des personnes désirant se marier ne correspondent pas parfaitement à la situation exacte. Nous voulons alors tenter d'engendrer de meilleurs résultats en analysant les autres modèles de choix. Avant d'étudier un second modèle, nous allons critiquer les résultats acquis avec les données d'entrainement. Il est effectivement possible d'obtenir de moyens résultats sur des données nouvelles si le modèle a tenté de s'adapter au mieux aux données qu'il a intégrées. Il n'est alors pas général et amène une situation d'overfitting. Il ne peut donc pas être utilisé pour d'autres situations. Nous doutons quand même de cette proposition car les données proviennent de la même population et suivent une même distribution avec peu de variables : la catégorie d'âges, le genre et le diplôme des individus.

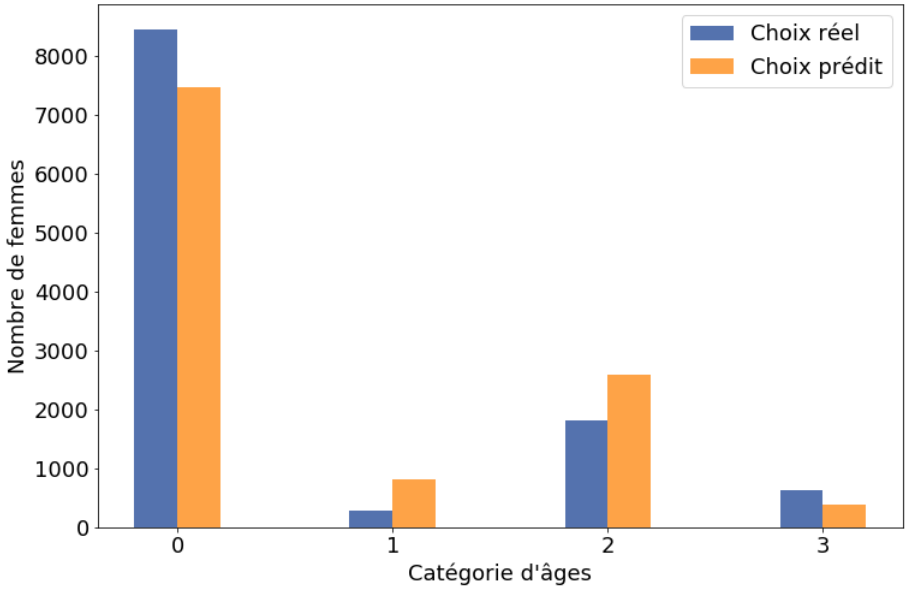
Le modèle utilisé est le même que précédemment, avec BIO comme algorithme d'optimisation. Il demande plus de ressources temporelles, justifié par un ensemble de données plus grand. Les dix simulations aléatoires s'exécutent en $38\,582$ secondes ou 10 heures 43 minutes et 2 secondes. Cela représente, en moyenne, 3858 secondes pour une simulation, soit 1 heure 4 minutes et 18 secondes. Nous n'affichons pas la progression du temps en fonction des simulations car le graphe est similaire au 3.5. Cette évolution est presque linéaire signifiant un temps d'exécution semblable entre les différentes itérations.

Nous avons ensuite calculé le nombre moyen de personnes entrant dans le marché des mariages sur les dix prédictions. Il vaut $52\,337$ contre $52\,262$ individus réellement mariés, soit un écart de 75 personnes. Ce nombre est faible sur la totalité des mariés réels puisque cela représente environ 1%. Le graphique 3.13 contient le nombre de personnes s'étant mariées au cours de l'année 2000 et sa prédiction à chacune des dix simulations. De nouveau, celles-ci oscillent autour du nombre réel. Nous retrouvons aussi une constance au niveau des prédictions. Ces dernières évoluent dans une courte échelle de valeurs, entre 51 900 et 52 800. Nous ne pouvons encore conclure sur la qualité de ce modèle avec les données d'entrainement. D'autres analyses doivent être encourues.

Nous avons construit les matrices de confusion standard et normalisée. Nous ne les affichons pas car, bien que les nombres au sein de la première matrice soient différents de la figure 3.9, la matrice de confusion normalisée est identique à la figure 3.10. Les conclusions tirées sont donc semblables. Notons tout de même que l'accuracy du modèle sur les données d'entrainement vaut $98,14\%$. Le choix discret n'est donc pas en situation d'overfitting. Il n'amène pas de précision supplémentaire au niveau des données utilisées pour le calibrer.



(a) Résultat concernant les hommes



(b) Résultat concernant les femmes

FIGURE 3.12 – Nombre de personnes au sein du marché des mariages par catégorie de diplômes pour le premier modèle de choix avec les données de test

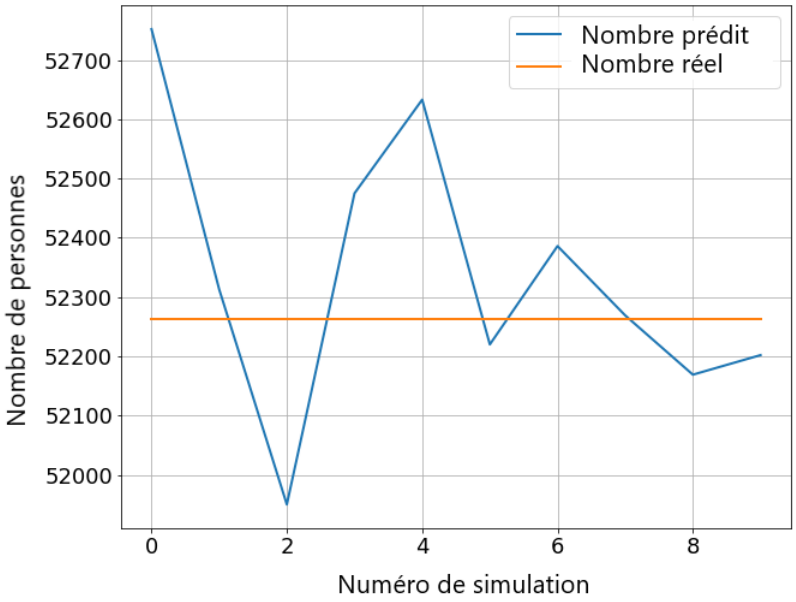


FIGURE 3.13 – Nombre de personnes désirant se marier par simulation pour le premier modèle de choix avec les données d'entrainement

Nous avons enfin engendré deux graphes en bâtonnets. De nouveau, il n'est pas nécessaire de les afficher car leur distribution est similaire à celles présentes à la figure 3.11. Nous retrouvons une croissance du nombre d'individus prédit, au niveau des deux genres, des classes 3 à 5 puis ensuite, une baisse exponentielle. Nous observons aussi que la classe d'âges $[20, 25[$ est trop peuplée dans le marché pour les hommes et pas assez pour leurs homologues féminins. Nous terminons par calculer les différences, pour les deux graphes, entre la distribution réelle du nombre d'individus s'étant mariés en 2000 et celle prédite. Pour les hommes, 3820 individus sont relevés. Chez les femmes, ce nombre est inférieur et vaut 3792. Ce premier modèle de choix peut être amélioré puisque les caractéristiques des personnes entrant dans le marché des mariages diffèrent de la réalité.

Nous avons alors réalisé trois autres modèles de choix discrets en vue de trouver une meilleure solution pour notre problématique. Tout d'abord, un autre algorithme d'optimisation est choisi : CFSQP. Il est conseillé de ne pas se concentrer sur un seul algorithme puisqu'ils peuvent apporter des résultats différents. En effet, comme explicité précédemment, un algorithme peut être coincé dans un minimum local. Nous avons effectué l'analyse de ce modèle avec le fichier de test. En effet, si les résultats obtenus avec ces données ne sont pas corrects, le modèle n'est pas assez général et n'est donc pas satisfaisant pour répondre à notre questionnement. Une vérification identique au modèle précédent est réalisée. Chaque itération demande, en moyenne, 1176 secondes, soit 19 minutes et 36 secondes. Les dix simulations s'exécutent alors en 3 heures 16 minutes et 6 secondes. De nouveau, l'évolution du temps d'exécution est presque linéaire.

Nous avons affiché, à la figure 3.14, les prédictions du nombre d'individus au sein du marché du mariage par itération. Nous retrouvons une oscillation dans une échelle de valeurs courte, de 22 900 à 23 300. Néanmoins, toutes les simulations proposent des nombres supérieurs à celui réel. Le nombre d'individus moyen dans le marché construit est de 23 066, soit un écart de 664 avec le nombre de personnes s'étant effectivement mariées au cours de l'année 2000.

Bien que le nombre de personnes au sein du marché soit plus important que le précédent, il n'est

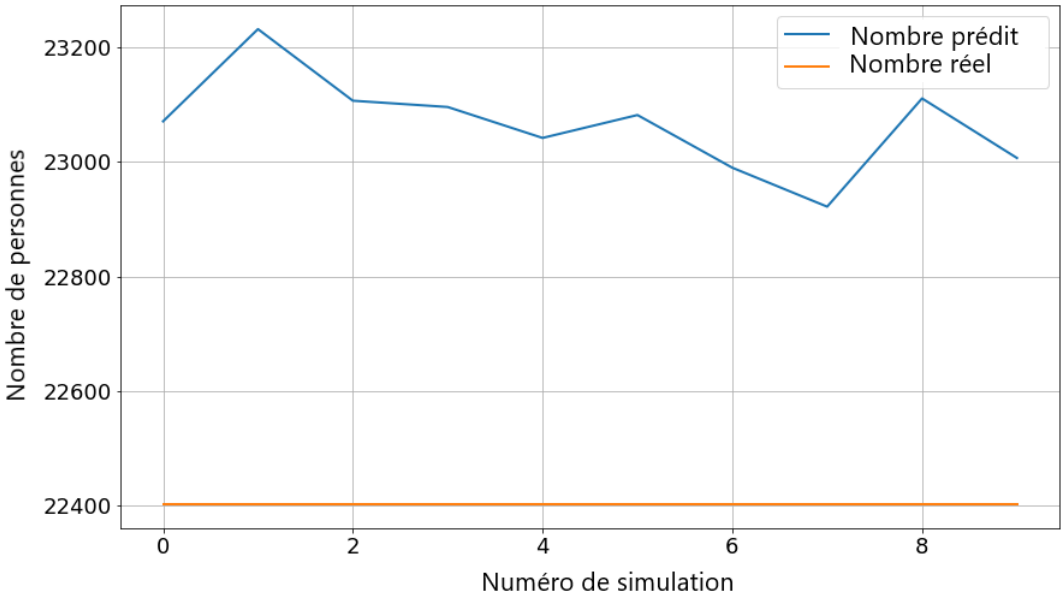


FIGURE 3.14 – Nombre de personnes désirant se marier par simulation pour le deuxième modèle de choix

pas intéressant d'afficher d'autres visualisations car les conclusions et analyses sont identiques à celles du premier modèle de choix. Discutons uniquement des différences de distributions du nombre d'individus réellement mariés en 2000 par catégorie d'âges et de la prédiction correspondante. Étant donné que nous avons réalisé cette étude par genre, deux chiffres sont exploités. Pour les individus masculins, une différence de 1500 personnes est obtenue en sommant les écarts pour chaque classe d'âges tandis que pour les femmes, ce nombre est plus élevé et vaut 1749.

Les deux modèles ne tenant pas compte des informations sur le diplôme des individus sont en tout point identiques. En effet, tant celui utilisant BIO que celui se reposant sur l'algorithme CFSQP retournent des coefficients de paramètres égaux. Nous effectuons alors une seule analyse. Le temps d'exécution moyen de ce modèle est proche des précédents, 20 minutes et 19 secondes. C'est la seule similitude avec les deux premiers modèles de choix. En effet, les dix prédictions renvoient, en moyenne, 3371 personnes dans le marché du mariage construit. Ce nombre est insuffisant puisque le nombre réel de mariages est 22 402. Nous concluons que la présence des données concernant les diplômes est essentielle pour obtenir un modèle de choix de qualité. Nous décidons de ne pas étudier ces deux dernières propositions puisque nous ne les retenons pas.

Concluons cette section en discutant de la validité de nos modèles de choix. Tout d'abord, il est important de conserver les informations relatives au diplôme des individus puisqu'en les écartant, les résultats obtenus ne sont pas satisfaisants. Le marché des mariages produit sans ces données est trop peu fourni, amenant des erreurs pour la suite de notre solution au problème initial. En effet, nous ne saurions engendrer un nombre adéquat de mariages si les individus à coupler manquent. Ensuite, les deux premiers modèles de choix discrets discutés n'amènent pas de mauvaises analyses. Les marchés produits contiennent un nombre similaire ou légèrement supérieur au total réel de personnes s'étant mariées en 2000. Néanmoins, les attributs de ces individus dans les prédictions ne correspondent pas tout à fait à la réalité. Cela provient peut-être de la faible quantité d'informations sur la population étudiée.

Pour obtenir une meilleure méthode, plus adaptée, offrant des distributions de caractéristiques simi-

laires à la réalité, nous avons envisagé d'étudier les réseaux de neurones. Nous pensons effectivement que ces modèles, plus complexes, peuvent capturer le manque d'informations. Toutefois, des inconvénients viennent avec cette complexité. Nous revenons sur ces propos à la section suivante.

3.2 Réseau de neurones

Cette section utilise uniquement la ressource [FRE2017], le cours que nous avons reçu à l'Université de Namur traitant de ce sujet. Les informations contenues dans celui-ci nous permettent de rédiger la section théorique ci-après.

3.2.1 Théorie

Un réseau de neurones artificiel est une architecture informatique qui tente d'imiter le comportement et les capacités d'un réseau de neurones biologique. Son objectif principal est d'être un outil performant pour résoudre différents problèmes informatiques complexes. Détaillons alors le fonctionnement d'un réseau de neurones à une seule couche puisque ces explications sont aussi valables dans le cas d'un réseau à plusieurs couches en tenant compte de quelques changements.

Schématisons tout d'abord un réseau de neurones basique. Notre représentation est une reproduction provenant de la ressource [FRE2017] et est observable à la figure 3.15. L'explication de ce dessin est réalisée grâce à une analogie avec les neurones cérébraux. Les unités d'entrée, x_j, sont semblables aux neurones sensoriels qui perçoivent une information particulière. Il faut distinguer x_0 qui représente uniquement une aide pour simplifier les notations. Tous ces neurones sont liés à une sortie par des connections appelées synapses. Ces dernières possèdent des poids, w_j, qui indiquent la force d'influence des x_j sur la sortie. Enfin, selon le résultat de la combinaison linéaire des entrées et des poids, le neurone de sortie est activé ou non, retournant certaines informations. Une fonction d'activation, $\sigma()$, doit alors être définie.

Les fonctions d'activation les plus courantes sont les suivantes :

- le modèle linéaire est celui possédant comme fonction d'activation $\sigma(a) = a$,

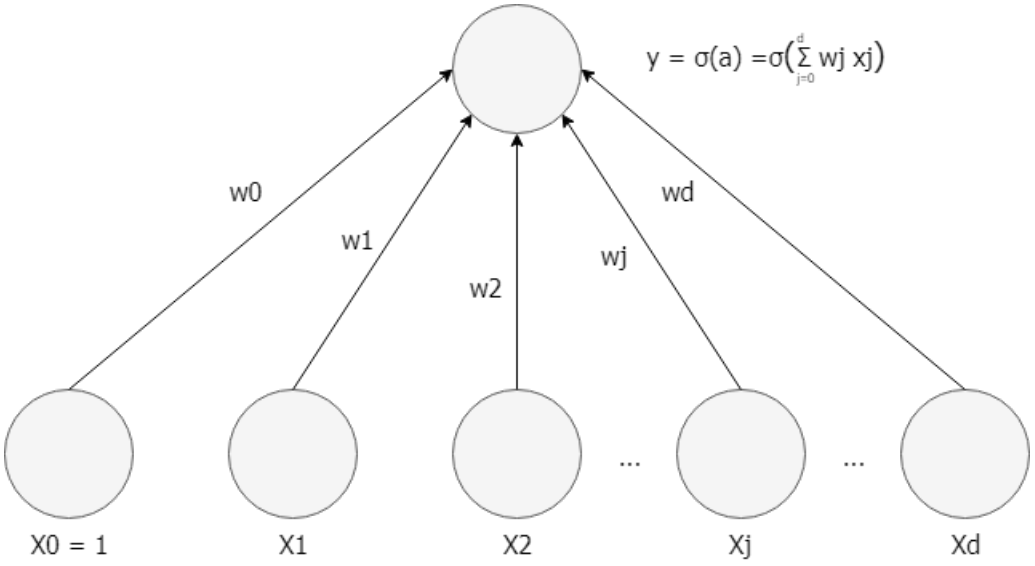


FIGURE 3.15 – Réseau de neurones théorique à une couche

- la sigmoïde $\sigma(a) = \frac{1}{1+\exp(-a)}$,
- le modèle linéaire discriminant $\sigma(a) = \text{sign}(a) = \begin{cases} 1 & \text{si } a > 0, \\ -1 & \text{sinon.} \end{cases}$

Nous les avons construites sur un graphe en fonction de la valeur du paramètre a. Celui-ci est visible à la figure 3.16.

Grâce aux réseaux de neurones, tant des classifications que des régressions peuvent être résolues. Ce sont ces premiers problèmes qui nous intéressent. En effet, nous voulons classer nos individus en deux groupes : celui des célibataires et celui des personnes désirant se marier au cours d'une année particulière.

L'algorithme d'un réseau de neurones pour résoudre un problème de classification en deux groupes est donc assez simpliste. Il contient une boucle while qui attend que tous les points soient bien classés, une boucle for qui analyse les individus un à un et une condition if qui vérifie la prédiction des instances. L'algorithme est décrit ci-après. Il utilise un discriminant linéaire comme fonction d'activation.

Entrée : ensemble de données $\mathcal{D} = (x_i, t_i)$
Sortie : poids w_t

Ajouter, devant chaque vecteur x, $x_0 = 1$
Initialiser les poids w_t aléatoirement

while au moins une instance est mal classée **:**

    for x_i **:**

        Calculer la prédiction $y_i = \text{sign } w_t^T x_i$

        if x_i est mal classée, $y_i \neq t_i$ **:**
            $w_{t+1} = w_t + \alpha_t(t_i - y_i)x_i$

return w_t

Bien qu'un réseau de neurones à une couche offre des performances remarquables, il est possible d'améliorer les résultats en ajoutant une ou plusieurs couches cachées. Un exemple est donné à la figure 3.17. Ajouter des couches de neurones occasionne la distribution du problème en plusieurs unités. Cela offre une complexité supplémentaire qui permet de résoudre des problématiques plus compliquées. Toutefois, cette solution possède aussi un inconvénient important : avec cette montée en complexité, il devient difficile de comprendre le processus interne au réseau.

Cet ajout de couches cachées était une solution à la crise des réseaux de neurones à une seule couche. En effet, ces derniers possédaient quelques limitations majeures dont notamment, l'incapacité de résoudre des problèmes de classification XOR. Les réseaux à plusieurs couches peuvent désormais approximer des fonctions arbitrairement compliquées. Leur complexité est définie par leur nombre de couches cachées mais aussi par le nombre de neurones au sein de ces couches.

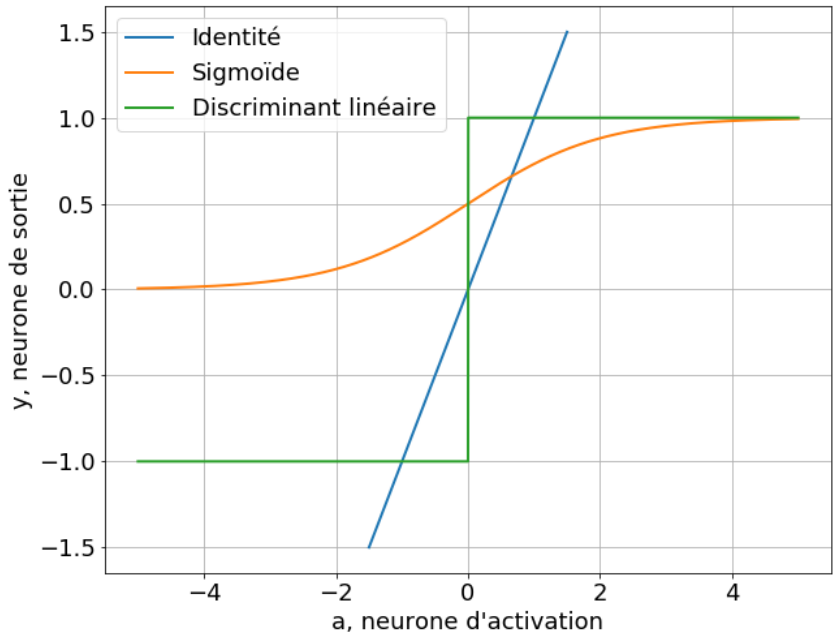


FIGURE 3.16 – Fonctions d'activation courantes

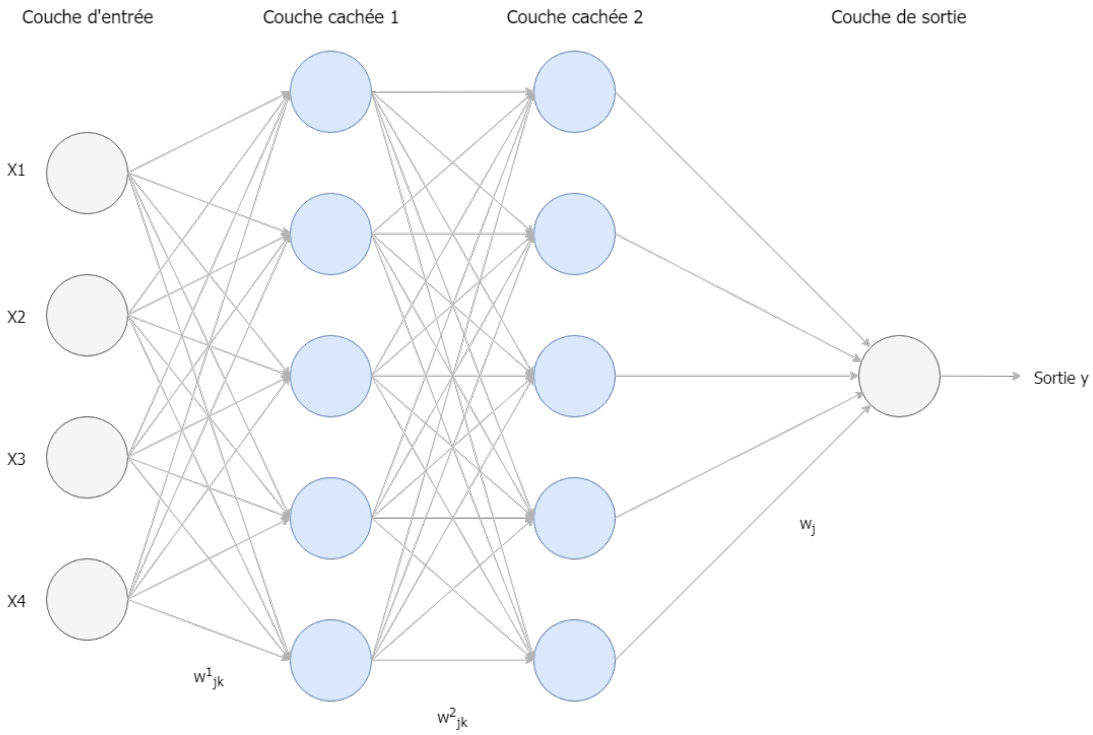


FIGURE 3.17 – Réseau de neurones théorique à plusieurs couches

Terminons cette section par justifier notre choix quant à l'utilisation de cette méthode. Tout d'abord, elle est assez facile à implémenter, spécialement en Python, qui possède des packages dédiés à la résolution de problèmes en Machine Learning. Chaque itération demande peu de ressources mais généralement, un nombre important d'étapes est nécessaire pour atteindre la convergence.

3.2.2 Implémentation

Avant de donner des explications sur notre implémentation, il nous semble important de justifier le choix du langage utilisé : soit Python 3. Premièrement, Python possède des packages importants et populaires contenant l'implémentation de réseaux de neurones et de fonctions utilitaires s'y rapportant. Ensuite, l'aide les concernant est bien fournie, la ressource internet `http://scikit-learn.org/stable/` contient des informations sur chaque fonction du package avec notamment les arguments d'entrée et de sortie ou encore des exemples bien complets. De plus, Python est un langage extrêmement populaire et un des plus recommandés pour le Machine Learning. Enfin, beaucoup de langages de programmation sont maintenant adaptés pour pouvoir inclure des bouts de code Python en leur sein. Le notebook Jupyter se référant à notre implémentation est disponible à l'annexe A.9.

Les données utilisées pour calibrer et vérifier notre modèle sont celles précédentes, d'entrainement et de test respectivement. Nous décidons de garder les données sur le genre et le groupe d'âges ainsi que la variable concernant les diplômes. En effet, le réseau de neurones comprend un processus de feature selection qui délaisse, sans supervision, les attributs non significatifs. Nous pouvons alors inclure les données sur le diplôme des individus en entrée de l'algorithme. Si elles ne sont pas nécessaires pour déterminer leur admission dans le marché, l'algorithme les écarte. Il est intéressant d'enlever des variables pour un réseau de neurones quand plusieurs d'entre elles sont inutiles. Cela permet de gagner un certain temps lors de la résolution de l'algorithme. Le réseau implémenté avec ces données est modélisé à la figure 3.18. Nous entrons donc la catégorie d'âges et de diplômes ainsi que le genre de l'individu dans le réseau. Après la propagation des informations dans les couches cachées, nous obtenons le choix de l'individu. La fonction d'activation de sortie est celle présentée auparavant, la sigmoïde. Celle associée aux neurones des couches cachées doit être définie dans l'algorithme. Par défaut, c'est relu, la fonction unité linéaire rectifiée, $f(x) = max(0, x)$.

L'algorithme choisit pour construire notre réseau de neurones provient de la ressource internet `http://scikit-learn.org/stable/`. Étant donné que nous réalisons une classification, nous nous sommes tournés vers la fonction MLPClassifier du package neural_network de ce site. Au vu de notre méconnaissance des paramètres du réseau, nous avons gardé les paramètres par défaut afin de tester

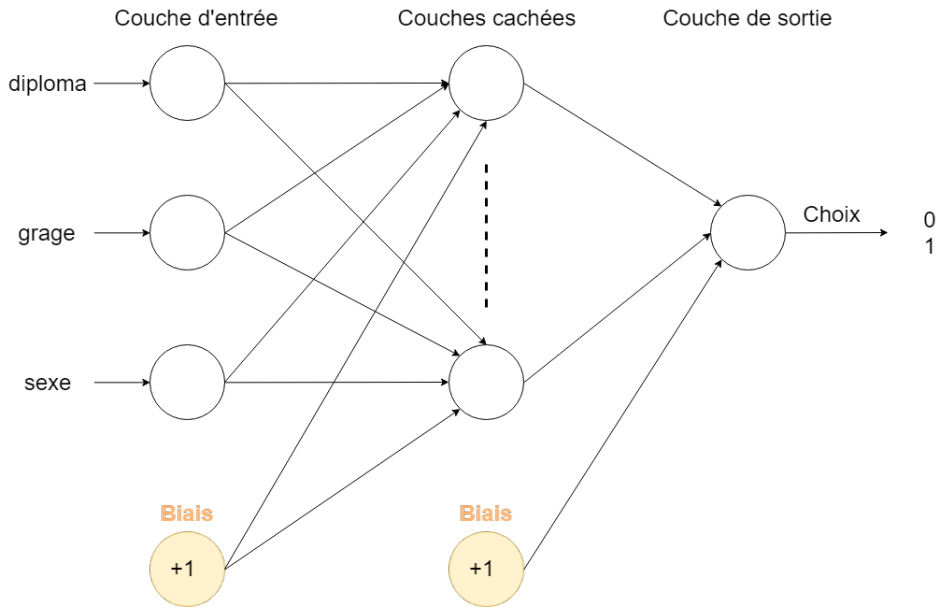


FIGURE 3.18 – Schéma du réseau de neurones construisant le marché des mariages

les différentes fonctions d'activation disponibles - identity, logistic, tanh et relu [1]. Notre objectif est de réduire cet ensemble pour conserver celles donnant les meilleurs résultats. Nous avons réalisé dix simulations avec dix architectures de réseau différentes pour chaque fonction d'activation. Pour nous aider dans notre étude, le tableau 3.1 affiche, pour chaque fonction, le temps moyen de résolution ainsi que le nombre moyen d'individus au sein du marché sur dix simulations. Le nombre réel de personnes s'étant mariées au cours de l'année 2000 est 22 402. Nous choisissons de conserver les fonctions d'activation relu et logistic puisqu'elles demandent moins de ressources temporelles et que leur marché contient un nombre moyen d'individus plus proche de la réalité. Notons que la fonction logistic porte aussi le nom de sigmoïde.

Nous avons ensuite analysé les différents solvers disponibles : lbfgs, sgd, adam. Le premier possède une meilleure performance sur les petits jeux de données alors que le dernier est conçu pour des datasets relativement grands en terme de temps de calcul tant pour l'étape d'entrainement que celle de test. Nous choisissons alors de continuer nos modèles avec le solver adam.

Ces deux paramètres discutés ainsi que la structure du réseau sont les plus importants dans l'algorithme. En conséquence, les autres restent ceux par défaut. Nous devons à présent choisir le nombre de neurones et le nombre de couches de ceux-ci afin de construire notre architecture. N'ayant que peu d'idées sur la situation idéale, nous avons utilisé la fonction GridSearchCV, implémentée en Python 3, qui repose sur la théorie de la sélection de modèles. Définissons celle-ci en mentionnant son utilité.

Commençons par documenter la sélection de modèles en explicitant sa définition et sa nécessité. Dans cet objectif, nous définissons les termes overfitting et underfitting. Le premier survient quand un modèle absurde est obtenu. Les données d'entrainement ont été apprises presque parfaitement mais la performance du modèle sur de nouvelles données est faible. Nous faisons face à une situation d'underfitting lorsque celui-ci ne modélise ni les données d'entrainement ni celles de test. Dans les deux situations, des performances médiocres de généralisation sont observées. La cause principale de cette constatation est le choix des méta-paramètres. À la différence des paramètres, ces derniers sont déterminés avant la période d'apprentissage du modèle et permettent d'obtenir des résultats, plus ou moins satisfaisants. La sélection de modèles revient à la sélection des méta-paramètres. Pour choisir le modèle offrant la meilleure prédiction, il faut déterminer le meilleur ensemble de méta-paramètres. Cette décision dépend évidemment des données, de leur quantité ou encore de leur qualité. Les méta-paramètres ne doivent pas être choisis manuellement. Plusieurs méthodes pré-définies dans différents langages de programmation sont disponibles à cet effet.

Pour élire la combinaison optimale de méta-paramètres, il est important de posséder un autre ensemble de données que celui d'entrainement puisqu'il est déjà connu du modèle. L'ensemble de test ne peut pas être employé puisqu'il permet de déterminer la qualité du modèle, après calibration. L'astuce est de séparer les données d'entrainement en deux ensembles : le premier portant le même nom et le second, l'ensemble de validation. Des techniques de validation existent pour que la séparation soit la plus op-

Fonction d'activation	identity	relu	tanh	logistic
Temps de résolution moyen (s)	4863	4289	4737	3482
Nombre moyen de personnes dans le marché du mariage	20 558	22 896	20 814	21 966

TABLE 3.1 – Indicateurs de comparaison associés aux différentes fonctions d'activation

1. Pour des informations théoriques sur ces fonctions, la ressource `http://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html` est disponible.

timale possible. Expliquons une stratégie conseillée, la k-fold cross-validation. Le jeu de données est divisé en k parties. L'algorithme est répété autant de fois qu'il y a de parts, soit k. À chaque répétition, l'ensemble d'entrainement est constitué de $k-1$ parties et celui de validation d'une seule. L'erreur estimée au final est une moyenne des erreurs à chaque répétition. Il est important de préciser que chaque partie est utilisée une et une seule fois comme validation.

La fonction Python GridSearchCV utilise cette technique de k-fold cross-validation pour déterminer le meilleur ensemble de méta-paramètres parmi la liste introduite. Pour chaque combinaison de méta-paramètres de l'ensemble initial, cette fonction réalise une cross-validation et calcule l'erreur associée. Ensuite, lorsque tous les assemblages ont été étudiés, elle retourne l'association de méta-paramètres offrant l'erreur minimale. Cette fonction a donc pour objectifs de valider le modèle proposé ainsi que de réaliser une sélection de modèles. Dans notre mémoire, nous utilisons la valeur $k = 10$ à chaque fois que la fonction GridSearchCV est exploitée puisque cette valeur est conseillée à la ressource [FRE2017].

Dans notre spécification du réseau de neurones, nous avons décidé d'appliquer le GridSearchCV uniquement sur le méta-paramètre qui concerne l'architecture du réseau. Nous allons donc réaliser deux réseaux, un pour l'activation relu et l'autre pour la fonction logistique. Nous pouvons intégrer cet autre méta-paramètre dans la sélection de modèles, cela est plus efficace. Notre choix est toutefois motivé par le gain de temps important. Pour calibrer le réseau de neurones en considérant uniquement trois architectures au sein de la fonction GridSearchCV, nous obtenons un temps moyen de 2856 secondes ou 47 minutes et 36 secondes sur dix simulations. La calibration de ce modèle en considérant deux méta-paramètres lors de la sélection de modèles, l'architecture et la fonction d'activation avec trois et deux possibilités respectivement, demande $20\,681$ secondes ou 5 heures 44 minutes et 41 secondes en moyenne pour cinq itérations. Pour explorer diverses pistes, cette dernière solution n'est pas envisageable selon nous. Pour déterminer la meilleure fonction d'activation, une analyse graphique est alors réalisée. Cette alternative, moins robuste, demande moins de ressources temporelles.

Finalement, une fois le modèle généré avec les meilleurs méta-paramètres, nous calculons les probabilités associées à l'ensemble de test afin de vérifier la qualité des modèles.

3.2.3 Vérification

Analysons maintenant les résultats obtenus par nos deux réseaux de neurones : le premier utilisant comme activation relu et le second, la fonction logistique. Cette sous-section est aussi disponible dans le notebook Jupyter à l'annexe A.9. L'étude se déroule de manière similaire pour les deux algorithmes. Nous commençons par discuter des méta-paramètres optimaux et de quelques indicateurs importants comme le temps d'exécution. Ensuite, plusieurs visualisations sont construites. Ce sont celles proposées lors de la vérification des modèles de choix discrets, à la section 3.1.4.

Avant de discuter de cette analyse, il est important de spécifier quelques détails techniques sur ces réseaux. Tous les deux voient leur algorithme d'optimisation converger. Néanmoins, il est impossible de connaitre le nombre d'itérations nécessaires en raison de l'utilisation de la fonction GridSearchCV. Cet usage amène une autre conséquence. Les poids des connections du réseau sont inaccessibles. Une solution à ces problèmes est l'implémentation de nos modèles sans l'utilisation de cette fonction avec les méta-paramètres optimaux. Néanmoins, nous n'obtiendrons pas les mêmes résultats puisque d'une part, la cross-validation n'est pas réalisée et d'autre part, l'aléatoire occupe une place importante dans nos modèles. Ensuite, le temps d'exécution de la vérification de ces deux modèles est identique puisqu'une même méthode est appliquée sur les probabilités sorties. Dix simulations ont été accomplies en 1199 secondes ou 19 minutes et 59 secondes, soit une moyenne de deux minutes par itération.

Commençons par discuter du premier réseau de neurones implémenté. Il utilise comme fonction d'activation, relu. Nous avons choisi de tester les valeurs du méta-paramètre lié à la structure du réseau par groupe de trois. C'est un compromis entre optimalité et temps de résolution. En effet, nous avions commencé en soumettant six structures différentes et l'algorithme utilisait trop de ressources temporelles, une moyenne de 3 heures 2 minutes 44 secondes sur cinq simulations contre 47 minutes et 36 secondes pour trois possibilités d'architecture.

Après différentes combinaisons, nous avons remarqué que dans toutes les situations expérimentées, les réseaux de neurones à une seule couche sont sélectionnés. Nous nous sommes concentrés sur ceux-ci en choisissant alors le nombre de neurones dans la première couche. Le dernier ensemble de méta-paramètres testé est le suivant : 26, 27 ou 28 neurones. Après la résolution de notre algorithme, il ressort que 26 neurones dans une seule couche retourne le moins d'erreurs dans la décision des individus. L'analyse qui suit est réalisée sur les données de test puisque l'objectif de cet ensemble est de vérifier la qualité du modèle.

Nous avons alors affiché le nombre de personnes s'étant mariées en 2000 et ses dix prévisions. Le graphique est disponible à la figure 3.19. Le réseau de neurones implémenté possède des prédictions oscillant autour du nombre réel. Sur ces dix simulations, il y a, en moyenne, 22 407 personnes dans le marché des mariages construit. Nous obtenons alors une faible différence avec le nombre réel, étant de 22 402. Néanmoins, nous pouvons observer que la dernière simulation, celle utilisée pour la suite de cette analyse, possède un marché plus fourni, avec 22 592 individus, soit un écart de 190 personnes.

Nous avons ensuite réalisé deux matrices de confusion, une standard et une normalisée, présentes aux figures 3.20 et 3.21. Sur la première, nous retrouvons le même constat qu'auparavant, beaucoup de célibataires décident de le rester et cela provoque une bonne mesure de l'accuracy, $98,23\%$. Nous observons aussi beaucoup d'individus mal-classés. De la matrice de confusion normalisée, nous concluons que les bonnes prédictions sont au nombre de 19% pour les personnes s'étant effectivement mariées. Néanmoins, ce résultat médiocre tire peut-être son explication d'une information non-prise en compte

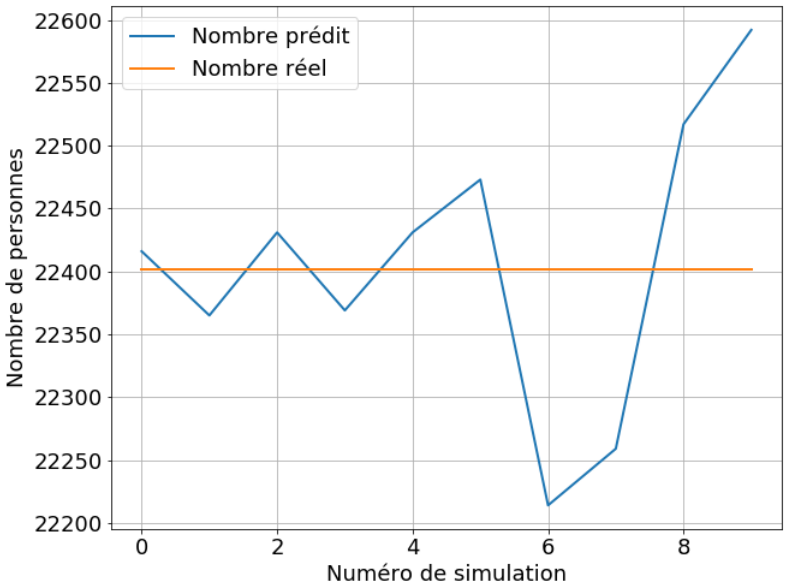


FIGURE 3.19 – Nombre de personnes désirant se marier en 2000 par simulation pour le réseau de neurones avec activation relu

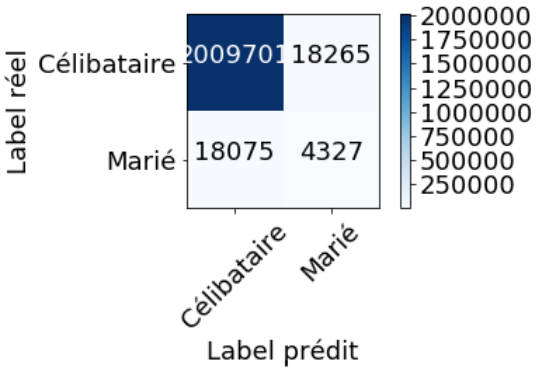


FIGURE 3.20 – Matrice de confusion pour le réseau de neurones avec activation relu

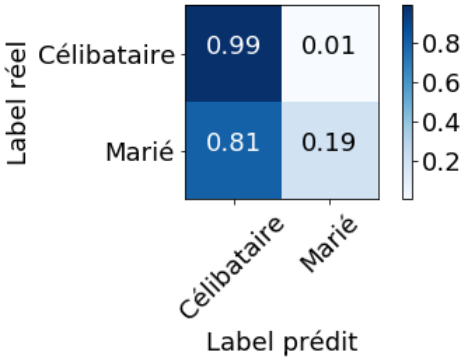
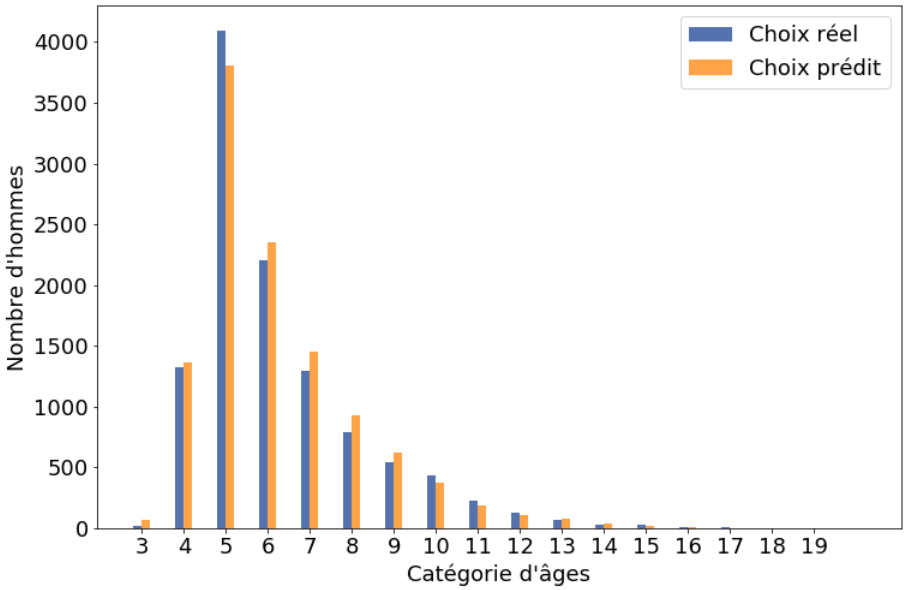


FIGURE 3.21 – Matrice de confusion normalisée pour le réseau de neurones avec activation relu

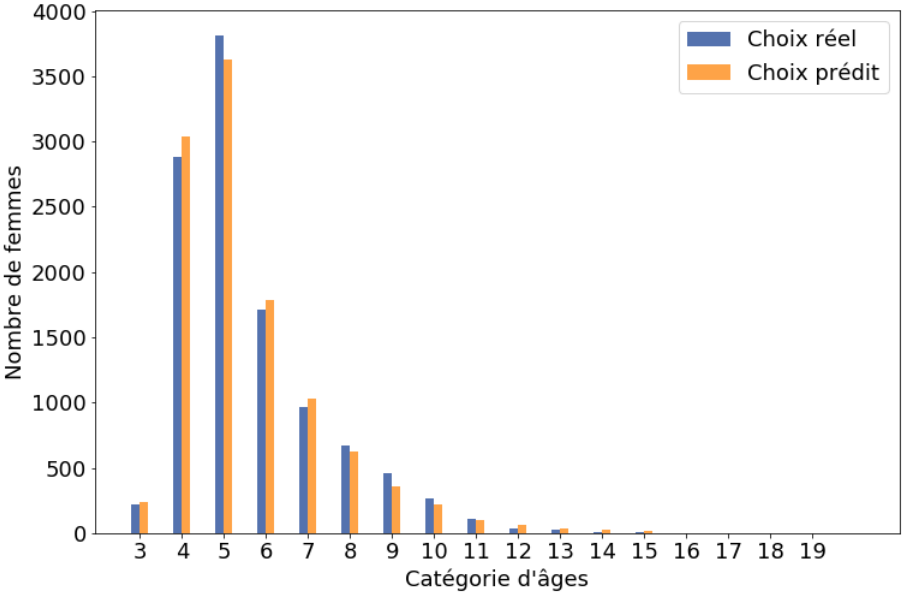
pour résoudre l'algorithme. Considérant seulement le groupe d'âges, le genre et le diplôme des individus, plusieurs personnes aux mêmes caractéristiques démographiques se trouvent dans notre jeu de données. Il est évident que parmi ces groupes de personnes partageant les mêmes attributs, nous retrouvons tant des mariés que des personnes célibataires en 2001. Or, celles-ci sont similaires. Cela peut donc expliquer le mauvais résultat observé à la matrice 3.21.

Pour vérifier cette intuition, nous avons étudié la distribution des personnes désirant se marier en distinguant leur genre, par groupe d'âges mais aussi par diplôme. Les graphiques s'y rapportant sont disponibles aux figures 3.22 et 3.23 respectivement. Au niveau des informations concernant la catégorie d'âges, nous retrouvons une distribution semblable à la réalité pour les prédictions tant pour les hommes que pour leur conjointe. Nous avons tout de même calculé la différence entre ces deux distributions. Elle vaut 1046 personnes pour les hommes et 788 pour les femmes. Suite à l'implémentation de notre algorithme dans une population synthétique, des résultats plausibles sont engendrés. Ce constat est similaire pour les graphes concernant le diplôme des individus avec pour le graphique des hommes, une différence de 754 et pour celui des femmes, 530. Nous concluons que malgré quelques différences, le marché du mariage créé contient des individus probables.

Nous avons aussi réalisé cette étude pour le réseau de neurones utilisant comme activation, la fonction logistique. Une même observation a été enregistrée : une seule couche de neurones offre le moins d'erreurs dans nos prédictions. Après avoir testé plusieurs triplets de nombres de neurones pour l'unique couche, nous nous sommes arrêtés à la soumission de ces structures : (24), (25), (26). Le nombre de neurones apportant moins d'erreurs de validation est 24. La calibration du modèle, avec cette fonction d'activation, demande, en moyenne, 1 heure 5 minutes et 28 secondes sur dix simulations. Cette durée est plus grande par rapport au temps de calibration du modèle reposant sur la fonction relu.

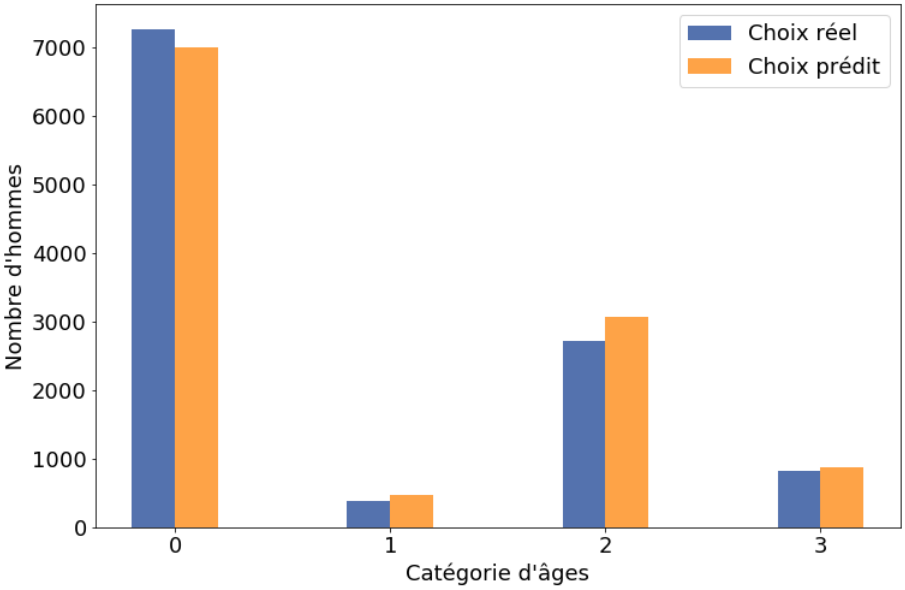


(a) Résultat concernant les hommes

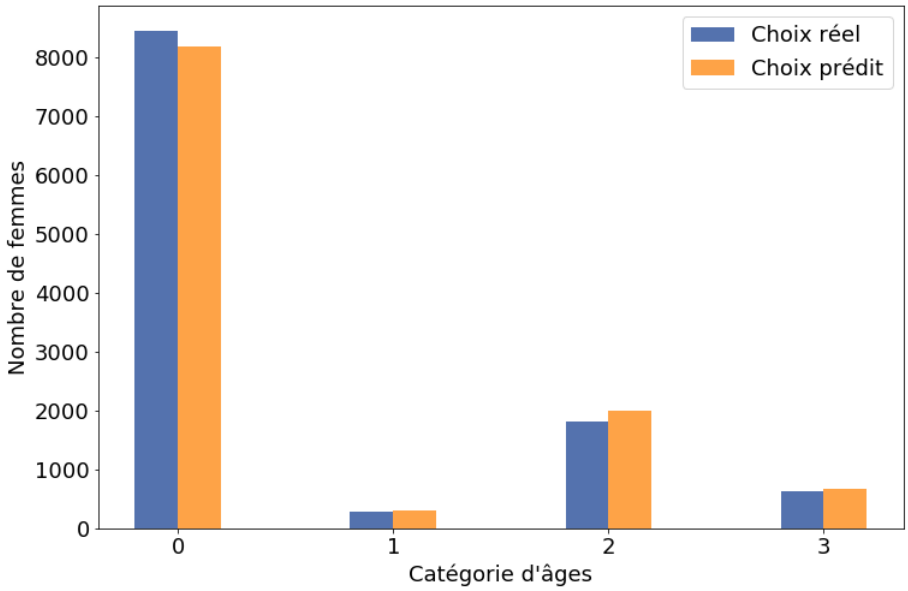


(b) Résultat concernant les femmes

FIGURE 3.22 – Nombre de personnes au sein du marché des mariages par catégorie d'âges pour le réseau de neurones avec activation relu



(a) Résultat concernant les hommes



(b) Résultat concernant les femmes

FIGURE 3.23 – Nombre de personnes au sein du marché des mariages par catégorie de diplômes pour le réseau de neurones avec activation relu

La première visualisation 3.24 analysant la qualité du modèle est un graphique linéaire représentant le nombre de personnes se mariant en réalité et en prévision, par simulation. De nouveau, les prévisions oscillent autour du nombre d'individus qui se sont effectivement mariés en 2000. Notre marché construit contient en moyenne $22\,351,5$ personnes sur les dix itérations. Nous obtenons alors un écart de $50,5$ avec les $22\,402$ personnes réellement mariées au cours de l'année 2000. Nous nous attendons à recevoir des analyses semblables au réseau précédent.

Nous avons, une nouvelle fois, construit deux matrices de confusion, une standard et une normalisée. Donnant des analyses identiques aux précédentes figures 3.20 et 3.21, nous ne les affichons pas. Terminons alors avec les distributions réelle et prédite des individus par groupe d'âges et par diplôme, pour les deux genres. De nouveau, obtenant une étude similaire à celle réalisée pour les figures 3.22 et 3.23 dûe à des distributions semblables, il n'est pas nécessaire d'afficher les graphes construits. Nous discutons uniquement des différences relevées sur ces graphes par catégorie d'âges et de diplômes. Lors de l'étude par âge des personnes entrant dans le marché, nous observons une différence de 1520 hommes et 752 femmes avec la réalité. Pour les classes de diplômes, ces nombres valent respectivement 234 et 84. Ce réseau capture donc mieux les caractéristiques démographiques des femmes. Nous remarquons également que ce modèle modélise mieux l'éducation des individus que le premier réseau. Il ne faut néanmoins pas oublier que nous travaillons avec de l'aléatoire et donc ces chiffres sont à utiliser avec précaution.

Pour conclure le point destiné aux réseaux de neurones, effectuons un rapide rappel sur la qualité de ces modèles. En effet, ils offrent de bonnes performances au niveau de la prédiction des choix des individus inconnus du modèle. De plus, une fois l'étape de calibrage effectuée, le calcul de la décision des individus est très rapide puisqu'il s'effectue en moyenne en 2 minutes. Précisons que nous pouvons continuer à améliorer les performances des réseaux de neurones en analysant d'autres structures ou d'autres méta-paramètres mais l'objectif de ce mémoire est d'obtenir une procédure robuste pour le

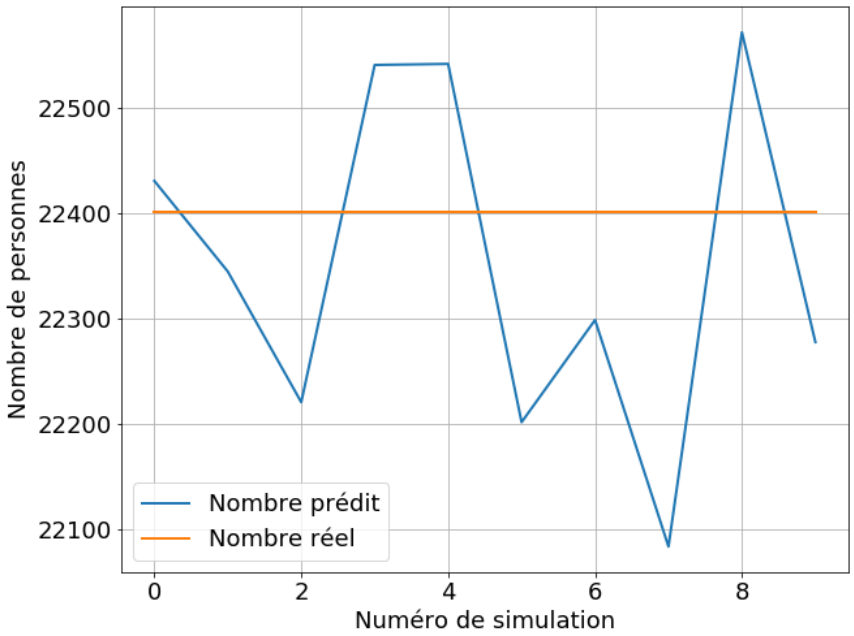


FIGURE 3.24 – Nombre de personnes désirant se marier par simulation pour le réseau de neurones avec activation logistique

module des mariages. Nous devons alors distribuer nos efforts sur deux parties et pas sur celle-ci uniquement. Pour finir, débattons rapidement sur la fonction d'activation à conserver. Toutes deux offrent des résultats assez similaires avec un nombre de personnes au sein du marché plus proche de la réalité pour logistique. De plus, cette dernière apporte de plus faibles différences au niveau des caractéristiques démographiques de la population désirant se marier. Nous conseillons alors, pour un futur travail dans ce domaine, d'utiliser cette fonction.

3.3 Conclusion et perspectives

Après avoir analysé en profondeur deux méthodes différentes, nous pouvons conclure sur leur qualité. Les deux types de modèles implémentés offrent une belle précision et prédisent le choix des individus avec peu d'erreurs. Tous deux peuvent généralement être insérés dans un code externe, dans un autre langage. En effet, les modèles de choix donnent simplement des coefficients d'utilité. Pour obtenir les probabilités et par conséquent, le choix des agents, un algorithme simple doit être écrit dans le langage de la population synthétique. Ses résultats peuvent alors être utilisés dans tout langage de programmation. Au niveau du réseau de neurones, nous pouvons également coder le réseau obtenu grâce aux poids reçus et plusieurs opérations simples. Néanmoins, ce travail est plus complexe. Une autre possibilité est de coder avec les mêmes méta-paramètres un réseau de neurones dans le langage de la population virtuelle. Enfin, plusieurs langages populaires offrent des modules permettant d'inclure un bout de code Python. Ensuite, le choix discret amène un avantage que ne possède pas le réseau de neurones. Il est facilement interprétable. En effet, les coefficients estimés nous permettent d'analyser le lien de chaque variable sur la décision des individus. Au contraire, le réseau de neurones à plusieurs couches et/ou plusieurs neurones est une boite noire. Nous ne pouvons pas interpréter facilement les calculs derrière les résultats retournés. Le dernier point de comparaison est le temps d'exécution mis par l'algorithme pour extraire le choix des individus. Une fois calibré, le réseau de neurones demande 2 minutes alors que le modèle de choix en utilise 20.

Nous concluons ce chapitre en annonçant qu'il est préférable de privilégier le réseau de neurones dans notre situation puisqu'il demande moins de ressources temporelles. De plus, les caractéristiques démographiques des personnes contenues dans le marché des mariages créé sont plus proches de la réalité pour le réseau de neurones. Cette méthode permet alors d'obtenir une base solide pour la deuxième étape de la dynamique du mariage : coupler les individus au sein du marché du mariage.

Chapitre 4

Couplage des individus au sein du marché des mariages

Après avoir créé le marché des mariages comprenant l'ensemble des individus désirant se marier, il faut à présent former les couples. Implémenter le matching des partenaires est la partie la plus complexe de notre solution de modélisation du processus des mariages. En effet, nous devons grouper les individus deux à deux tout en respectant leurs critères de choix de partenaires. N'ayant pas d'indication sur la meilleure démarche à suivre, nous avons voulu explorer deux solutions proposées dans la littérature. Nous avons pensé nos modèles de manière à trouver des couples suivant la même distribution que dans nos données.

Nous avons débuté ce chapitre en définissant, au moyen de deux méthodes, les préférences des hommes envers chaque femme de l'ensemble de choix. Cette étape est nécessaire car nos données contiennent uniquement des informations sur le choix des individus et non pas sur les attentes envers la conjointe idéale. Or, ces données sont cruciales pour la section suivante qui concerne l'implémentation des modèles de couplage. Nous exploitons d'abord une solution probabiliste. Cette dernière est travaillée avec plusieurs variantes. Ensuite, un problème d'optimisation combinatoire est résolu grâce à la méthode du recuit simulé. Ces deux idées différentes sont analysées et comparées pour n'en conserver qu'une seule. La dernière section comprend l'implémentation de cette solution sur le marché des mariages construit au chapitre précédent. Nous voulons, en effet, vérifier nos résultats pour conclure sur la qualité de ces méthodes.

Sommaire

4.1 Définition des préférences

Les algorithmes de couplage implémentés à la section 4.2 nécessitent la spécification des préférences des individus parmi les célibataires du sexe opposé. Nous avons décidé, dans notre mémoire, pour éviter la complexité, que seuls les hommes choisissent leur partenaire. Nous obtenons alors un modèle homme-dominant. Ce choix est purement arbitraire, nous pouvions prendre la décision opposée et produire un modèle femme-dominant. Nous n'avons pas implémenté cette solution car nous pensons que les résultats obtenus seraient similaires et que les conclusions tirées pourraient être adaptées aux femmes. Nous avons donc séparé les hommes et les femmes dans deux jeux de données différents. De plus, nous avons créé, pour chaque ensemble, des données d'entrainement et de test.

Pour analyser les préférences au sein des couples, nous avons considéré les personnes qui se sont mariées entre 1991 et 2000, soit une période de dix ans. En effet, nous pensons que des mariages plus anciens ne s'appuyaient pas sur les mêmes critères de choix. Conserver une seule année semble plus correct puisque les individus ont changé de catégorie d'âges sur dix ans. Nous voulions toutefois posséder une plus grande quantité de données pour augmenter en précision dans la modélisation. De plus, nous pensons que l'importance se situe au niveau de la différence d'âges. Si nous observons des résultats aberrants, nous nous concentrerons sur une seule année, à savoir 2000. Le notebook Jupyter ayant pour but de préparer les données et de réaliser les modifications décrites est disponible à l'annexe A.10.

Afin de définir les préférences des hommes parmi les femmes voulant se marier, nous avons décidé de sortir des probabilités de choix uniquement au niveau du groupe d'âges des individus. En effet, nous possédons une seule autre variable, celle concernant les diplômes. Toutefois, grâce à l'analyse réalisée au chapitre 2, nous pensons que cette information n'apporte pas de renseignement supplémentaire puisqu'une majorité des diplômes est indéterminée. En conséquence, choisir son épouse revient à choisir la catégorie d'âges à laquelle elle appartient.

Les probabilités désirées ont été générées par deux méthodes différentes, celle des choix discrets et celle des réseaux de neurones. Nous avons utilisé cette dernière méthode suite à ses bons résultats au chapitre 3. Nous avons également décidé d'étudier de nouveau les choix discrets puisqu'ils sont populaires dans la littérature. De plus, leurs résultats au chapitre précédent n'étaient pas médiocres. Nous ne rappelons pas la théorie concernant ces deux méthodes.

4.1.1 Méthode des choix discrets

Nous débutons par la méthode des choix discrets. La spécification du modèle pour le logiciel Biogeme est située à l'annexe A.11. Plusieurs fonctions d'utilité ont été implémentées, une pour chaque classe d'âges. En effet, nous voulons déterminer pour chaque homme, les préférences d'épouses au niveau des classes d'âges. Suite à cette étape, nous pouvons leur définir un ordre de prédilection pour chaque personne dans le marché du mariage féminin.

Après avoir calibré le choix discret, nous obtenons comme fonctions d'utilité :

U3 = 16,3 - 1,05 grage
U4 = 16,5 - 0,45 grage
U5 = 12 + 0,62 grage
U6 = 5,95 + 1,63 grage
U7 = 2,37 grage
U8 = -4,16 + 2,81 grage
U9 = -7,62 + 3,14 grage
U10 = -10,7 + 3,42 grage
U11 = -14,3 + 3,69 grage
U12 = -17,6 + 3,91 grage
U13 = -22,1 + 4,2 grage
U14 = -27,2 + 4,5 grage
U15 = -31,3 + 4,75 grage
U16 = -34,4 + 4,89 grage
U17 = -39,5 + 5,14 grage
U18 = -50,5 + 5,68 grage
U19 = -37,8 + 4,84 grage
U20 = 0

Nous avons fixé celle liée à la catégorie d'âges 20 au nul. La justification vient de l'importance de la différence entre les utilités et non pas de ces dernières en elles-mêmes. Ensuite, ce modèle de choix est spécifique et non plus générique, les coefficients à estimer sont spécifiques à chaque alternative. Bien que le modèle générique soit moins complexe, il n'est pas adapté à cette situation. Nous n'abordons pas le critère du log-vraisemblance car nous avons implémenté un seul modèle. Nous ne pouvons donc pas comparer cette valeur avec une autre. De plus, ne possédant pas de borne, son interprétation reste floue. Nous pouvons néanmoins discuter du ρ^2 ajusté. Il vaut $0,54$. C'est une valeur moyenne qui n'indique pas une bonne correspondance entre les données et les prévisions. Sans analyse approfondie, cet indicateur ne donne aucune conclusion sur l'aptitude du modèle puisqu'une justification cohérente peut expliquer sa faible valeur. Étudiant uniquement le groupe d'âges des individus, beaucoup d'hommes possèdent les mêmes caractéristiques mais n'ont pas effectué le même choix. Prenons un exemple concret pour détailler ce constat. Soit deux hommes âgés de 25 à 30 ans, Mr A et Mr B. Nos données ne contiennent aucune information pouvant les distinguer. Toutefois, Mr A a épousé Mme X, âgée de 25 à 30 ans et Mr B, Mme Y appartenant au groupe des 35 à 40 ans. Si nos algorithmes associent Mme Y à Mr A et Mme X à Mr B, la valeur du ρ^2 diminue puisque les prédictions sont fausses. Or, pour notre situation, cela n'a peu d'importance puisqu'au final, nous obtenons la même distribution.

Plusieurs visualisations ont alors été réalisées pour approfondir l'étude de la qualité du modèle obtenu. Le code Python utilisé pour cette partie est disponible à l'annexe A.12. Tout d'abord, pour prédire le choix des hommes, le modèle calibré utilise, en moyenne sur dix simulations, 72 secondes, soit 1 minute et 12 secondes. Comme indicateur de qualité, nous avons calculé, à chaque itération, la somme de la valeur absolue de la différence entre les choix réels et prédits des hommes pour chaque catégorie d'âges des conjointes. En moyenne, nous obtenons un écart de $1831,8$ personnes. Expliquons quelque peu cet indicateur. Nous observons, pour chaque classe d'âges, la différence entre le nombre d'hommes dont l'épouse appartient à cette catégorie et le nombre d'hommes dont la prédiction retourne ce choix. Nous prenons ensuite la somme de la valeur absolue de ces écarts pour quantifier la distance entre les deux distributions en question : la première concernant les conjointes réelles et la seconde indiquant les prédictions de notre modèle.

Nous avons alors affiché, à la figure 4.1, ces distributions pour la dernière itération afin d'analyser les différences relevées. Pour cette dixième simulation, l'écart calculé vaut 1864. L'histogramme indique que les deux distributions sont assez similaires. La différence la plus importante se situe au niveau de la catégorie d'âge 5. Le modèle de choix ne prédit pas assez de préférences pour ce groupe par rapport au nombre de conjointes réelles. Néanmoins, la similitude de distributions amène une constatation positive pour la section suivante. En effet, cela permet d'affirmer qu'une majorité des couples peut être formée au sein du marché des mariages en considérant le premier choix de l'homme.

Nous avons, ensuite, représenté les informations de ce graphe d'une manière différente. Nous avons alors construit une boite à moustaches, présente à la figure 4.2. Les différents indicateurs repris dans celle-ci - quartiles, médiane - sont similaires. Nous remarquons la présence d'outliers pour les classes supérieures à 10. Cette observation est justifiée par la présence peu nombreuse dans le marché de femmes de cette catégorie d'âges. Cette visualisation appuie l'analyse précédente. La distribution des prédictions de choix des hommes est similaire à celle des catégories d'âges de leur épouse réelle. En effet, nous ne notons aucune différence entre les deux boites à moustaches.

L'histogramme global visible à la figure 4.1 a été reproduit pour chaque catégorie d'âges d'hommes afin d'acquérir une analyse plus détaillée. Ces graphiques sont disponibles à la figure 4.3. Tout d'abord, il faut être vigilant à l'échelle de chaque graphique, qui diffère fortement. Cela appuie l'analyse des outliers réalisée précédemment. Ensuite, les distributions réelle et prédite de chaque graphique possèdent des allures comparables. Néanmoins, plusieurs différences importantes sont relevées. Généralement, moins de mariages extrêmes sont observés dans les prédictions. Nous émettons cette constatation suite à l'étude de l'étendue des distributions réelle et prédite. Uniquement pour les hommes âgés de 15 à 30 ans, soit trois catégories, nous possédons des préférences dans des catégories d'âges plus extrêmes qu'en réalité. Selon notre modèle, certains de ces individus masculins ont choisi des femmes de classe d'âges 3 alors que dans nos données, aucun mariage n'est existant entre ces personnes concernées.

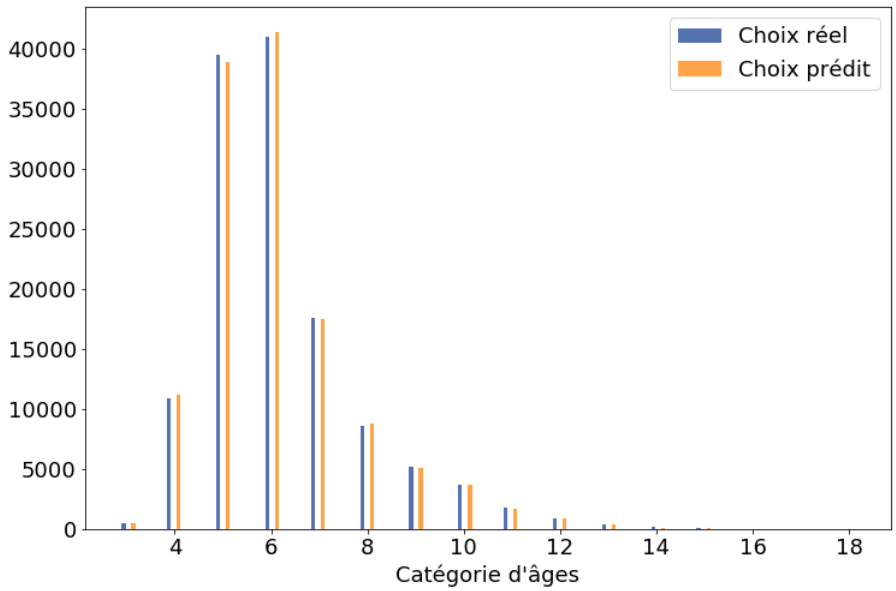


FIGURE 4.1 – Distribution des hommes selon la catégorie d'âges choisie pour le modèle de choix

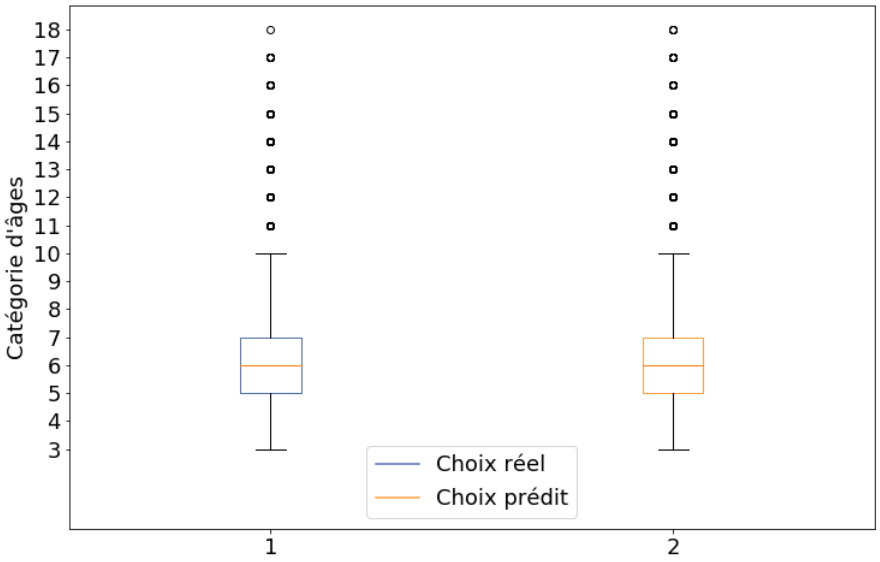


FIGURE 4.2 – Boite à moustaches de la catégorie d'âges choisie par les hommes pour le modèle de choix

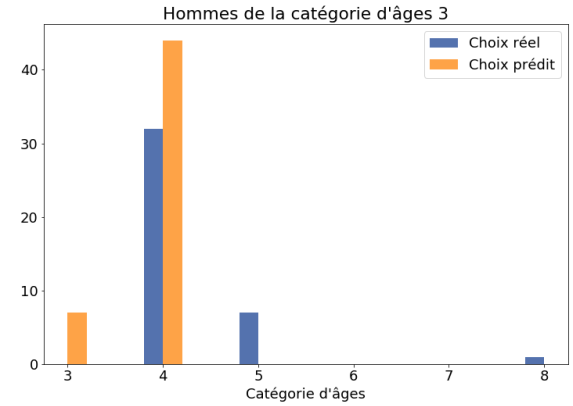
Nous avons ensuite affiché les relations entre l'âge des hommes et leur choix sur le graphique 4.4. L'image (a) contient les liens réels entre les partenaires au niveau des classes d'âges alors que le graphe (b) représente la relation entre l'âge des hommes et leur préférence selon le modèle de choix. Les deux visualisations possèdent des allures semblables. Nous remarquons toutefois que le modèle de choix retourne peu de prédictions extrêmes, où la préférence des hommes se dirige vers des femmes à la catégorie d'âges éloignée de la leur. Ces deux graphiques appuient de nouveau la faible présence d'hommes et de femmes âgés dans nos données.

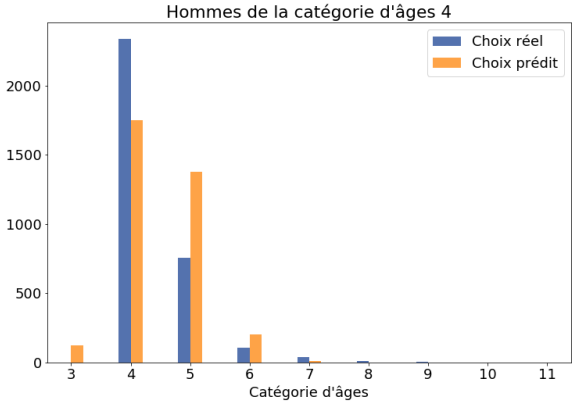
Enfin, nous avons voulu étudier la distribution des différences d'âges entre l'homme et sa conjointe réelle ainsi que celle préférée. Nous avons alors affiché un histogramme à la figure 4.5. Nous observons certaines différences dans les distributions. Néanmoins, les prédictions restent cohérentes. Nous n'obtenons pas de préférence extrême, la différence d'âges entre l'homme et son choix n'est pas élevée. L'écart le plus important entre les deux distributions se situe au niveau 0. Les préférences des hommes pour des femmes ayant la même catégorie d'âges qu'eux ne sont pas assez présentes dans les résultats du modèle.

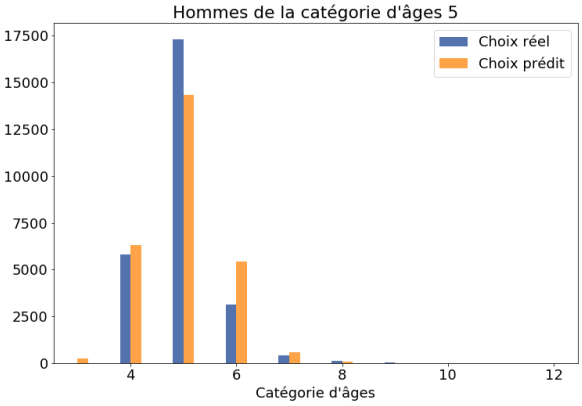
Concluons ce point sur le modèle de choix discrets implémenté. Celui-ci donne des résultats assez probables puisque nous relevons peu de préférences extrêmes. Néanmoins, certaines distributions diffèrent. Nous voulons donc explorer une autre solution afin d'obtenir un modèle offrant de meilleurs résultats, des prédictions plus proches de la réalité. Nous avons, en conséquence, effectué ce même choix discret avec comme algorithme CFSQP. Nous obtenions des analyses similaires, il n'est pas nécessaire d'en discuter davantage. Nous exploitons alors, à la section suivante, la théorie des réseaux de neurones. Cette solution peut être préférable si les méta-paramètres sont choisis de manière optimale.

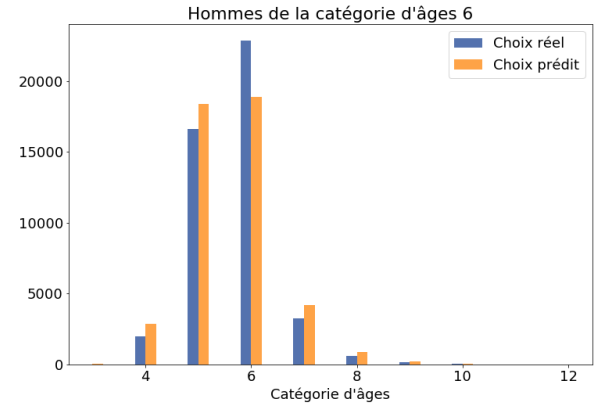
4.1.2 Résolution par les réseaux de neurones

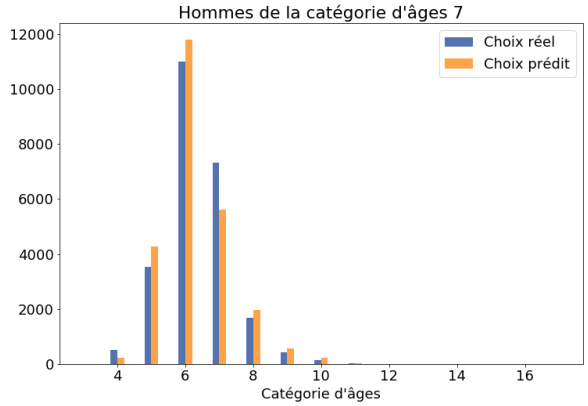
Étant donné notre satisfaction envers le réseau de neurones à la section 3.2, nous avons décidé d'utiliser, de nouveau, cette théorie afin d'obtenir un modèle offrant une meilleure qualité que le choix discret précédent. Le notebook Jupyter contenant son code et les analyses réalisées se trouve à l'annexe A.13.

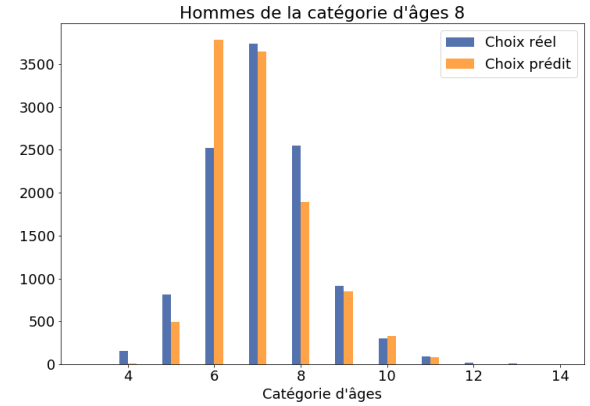

Hommes de la catégorie d'âges 3
Choix réel
Choix prédit
Catégorie d'âges

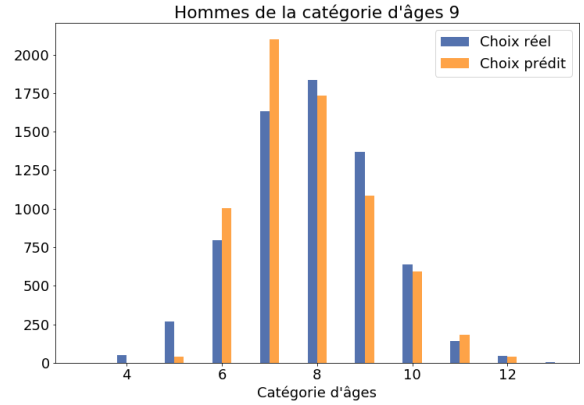

Hommes de la catégorie d'âges 4
Choix réel
Choix prédit
Catégorie d'âges

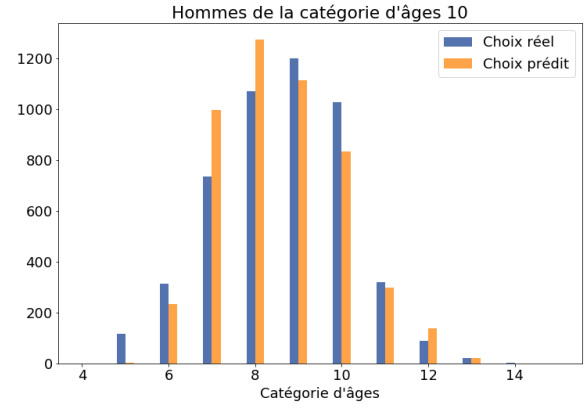

Hommes de la catégorie d'âges 5
Choix réel
Choix prédit
Catégorie d'âges


Hommes de la catégorie d'âges 6
Choix réel
Choix prédit
Catégorie d'âges


Hommes de la catégorie d'âges 7
Choix réel
Choix prédit
Catégorie d'âges


Hommes de la catégorie d'âges 8
Choix réel
Choix prédit
Catégorie d'âges


Hommes de la catégorie d'âges 9
Choix réel
Choix prédit
Catégorie d'âges


Hommes de la catégorie d'âges 10
Choix réel
Choix prédit
Catégorie d'âges

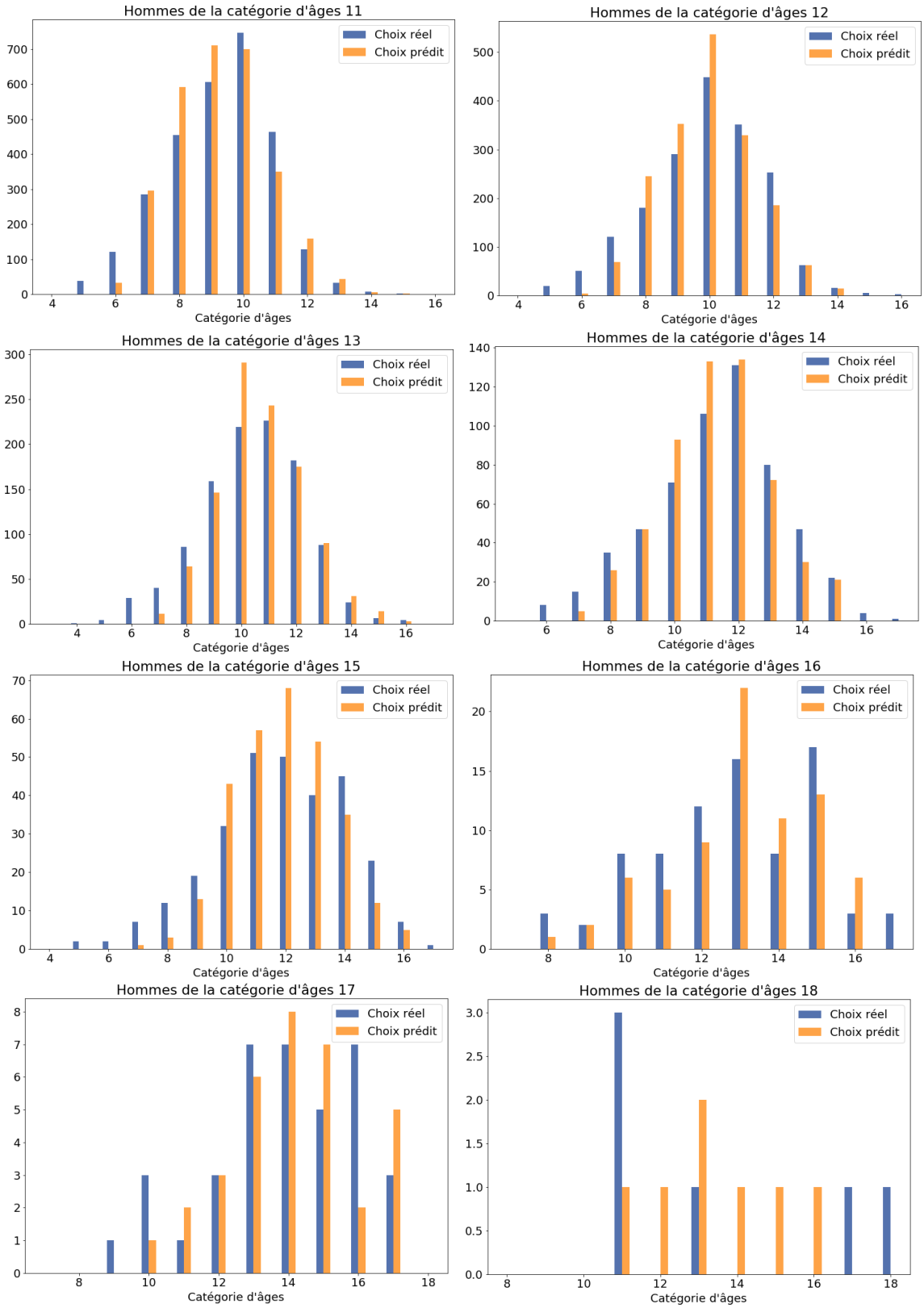
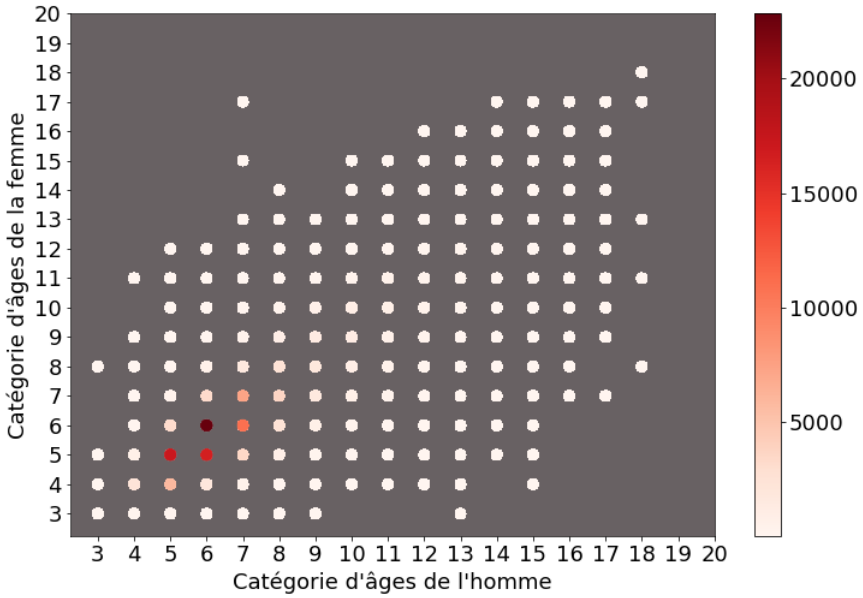
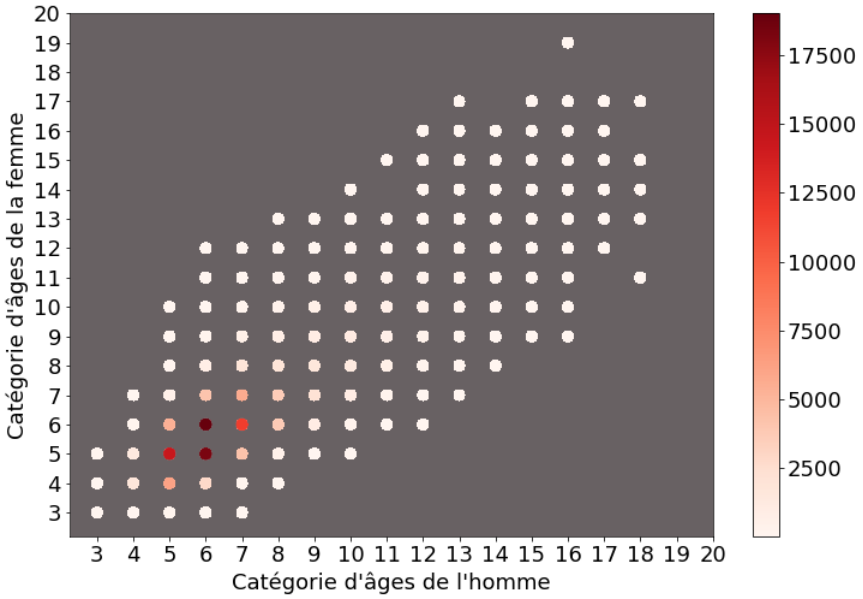


FIGURE 4.3 – Distribution des hommes selon la catégorie d'âges choisie par groupe d'âges masculin pour le modèle de choix



(a) Relation entre la catégorie d'âges des époux réels



(b) Relation entre la catégorie d'âges de l'homme et sa prédiction

FIGURE 4.4 – Relation entre l'âge des hommes et leur préférence pour le modèle de choix

Dans un premier temps, décrivons le réseau implémenté grâce à un schéma présent à la figure 4.6. Nous injectons une seule information en entrée du réseau, le groupe d'âges de l'homme considéré. Après quelques combinaisons de calculs élémentaires à l'intérieur des couches cachées, nous obtenons les préférences de l'individu pour les 17 catégories d'âges, de 3 à 19 car aucune femme de classe 20 n'est présente dans nos données. Cela se modélise par 17 neurones de sortie qui, suite à l'application de la fonction d'activation softmax, retournent les 17 probabilités attendues. Cette fonction est une exponentielle normalisée, une généralisation de la logistique. Grâce à un vecteur d'entrée réel à k dimensions, elle retourne k nombres réels strictement positifs dont la somme vaut 1.

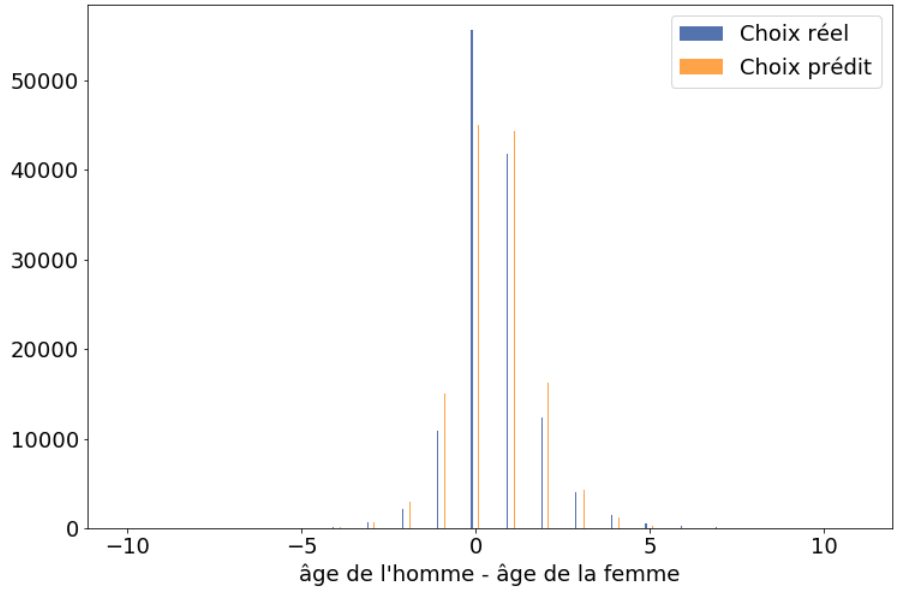


FIGURE 4.5 – Distribution des différences de catégorie d'âges entre l'homme et sa préférence pour le modèle de choix

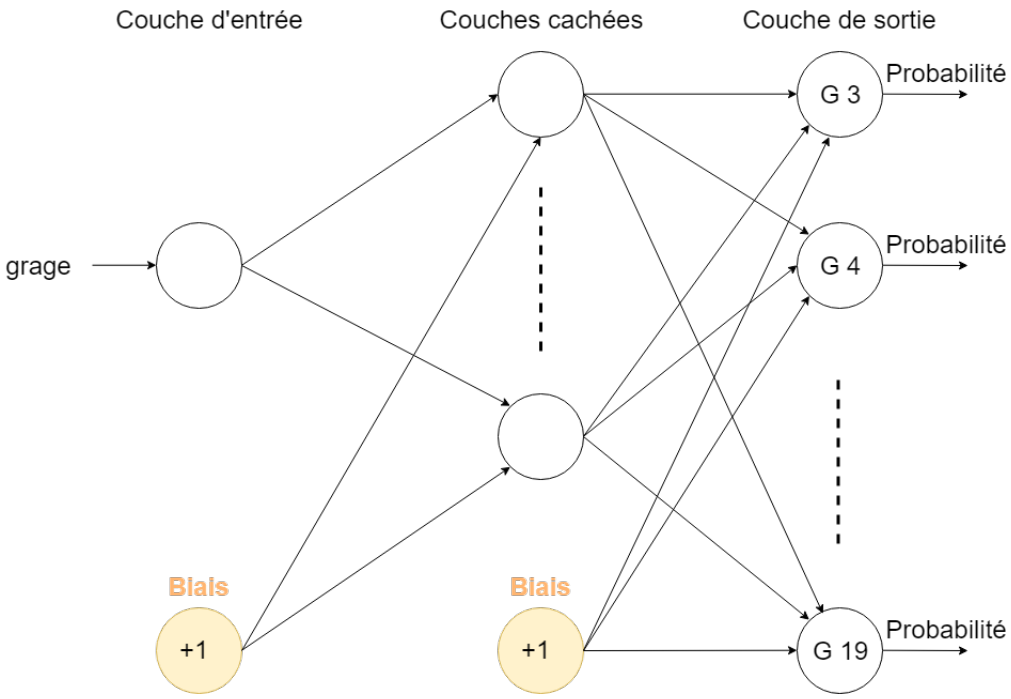


FIGURE 4.6 – Schéma du réseau de neurones déterminant les préférences des hommes

Sa définition est la suivante,

$$\sigma(x)_j = \frac{e^{x_j}}{\sum_{i=1}^{k} e^{x_i}} \quad \forall j \in \{1, \ldots, k\}\,.$$

Comme explicité précédemment, la fonction d'activation des couches cachées doit être définie par nos soins dans l'algorithme. Nous choisissons de conserver celle par défaut, relu, puisqu'elle fournit gé-

néralement de bons résultats. De plus, elle est très populaire dans tous les algorithmes de machine learning.

De manière similaire au premier réseau de neurones, nous utilisons comme solver adam puisqu'il est adapté aux grands jeux de données. Le dernier paramètre important à étudier est l'architecture du réseau. Nous appelons, de nouveau, la fonction GridSearchCV qui sélectionne, parmi un ensemble donné, les méta-paramètres engendrant l'erreur la moins élevée sur les données de validation. Cette cross-validation est une k-fold où k est fixé à 10. Pour ce modèle, nous proposons cinq structures à chaque simulation. Le dernier essai est le suivant $\{(30,), (25, 25), (40, 40, 40), (60,), (50, 50)\}$. L'architecture retournée par GridSearchCV est le couple $(25, 25)$. Nous avons affiché l'accuracy de ce modèle pour l'ensemble d'entrainement, soit $0,47$. Cette valeur est faible mais peut s'expliquer, comme précédemment, par le grand nombre d'hommes aux caractéristiques semblables mais au choix différent.

L'étape suivante est la vérification du réseau obtenu sur les données de test. Le calcul des choix des individus masculins grâce aux probabilités obtenues se déroule en 10 secondes, valeur moyenne reçue après dix simulations. À chaque itération, nous décidons de calculer également un indicateur de qualité du modèle. Nous chiffrons la différence des distributions réelle et prédite à propos du choix de catégorie d'âges effectué par les hommes. Pour chaque classe d'âges de conjointes, nous mesurons l'écart entre le nombre de personnes ayant effectivement une épouse de cet âge-là et le nombre d'hommes dont les prédictions placent leur préférence dans cette catégorie. En moyenne sur les dix simulations effectuées, cet indicateur vaut 3836. Il est supérieur à celui trouvé avec notre modèle de choix mais est néanmoins peu élevé.

Pour étudier la qualité de notre modèle, nous avons implémenté un ensemble de visualisations. Ces dernières sont similaires à la section antérieure. Commençons donc par analyser la distribution des choix effectués par les hommes du marché des mariages. Nous avons à disposition un histogramme ainsi qu'une boite à moustaches, aux figures 4.7 et 4.8 respectivement. Ces deux graphes affichent des similarités entre la réalité et les prédictions. Les deux boites à moustaches sont identiques. Cette constatation a déjà été relevée pour le modèle de choix. Quelques différences sont remarquables dans l'histo-

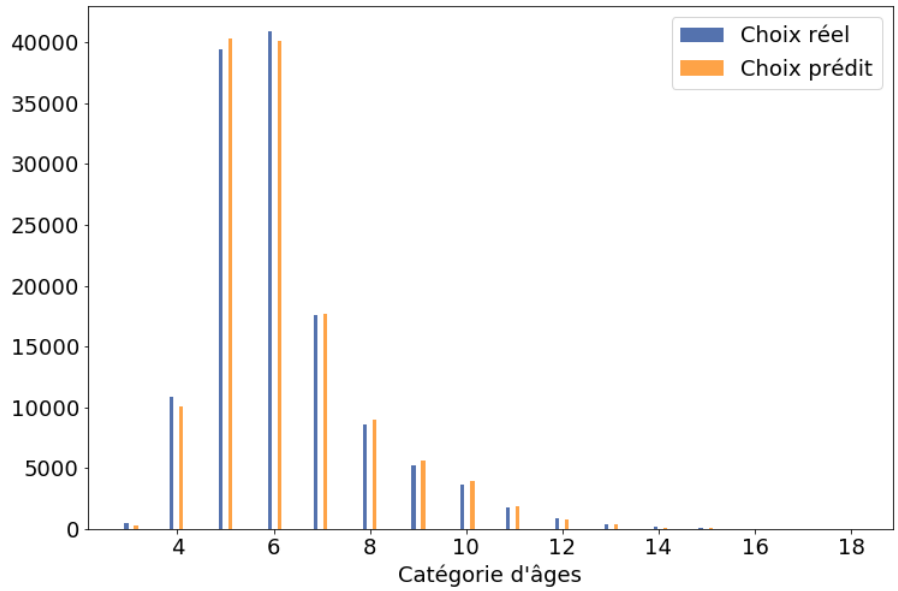


FIGURE 4.7 – Distribution des hommes selon la catégorie d'âges choisie pour le réseau de neurones

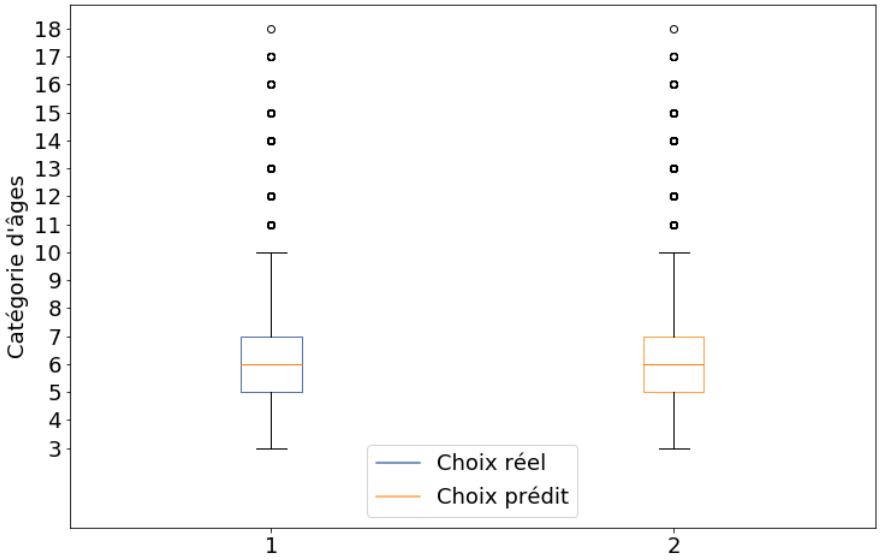


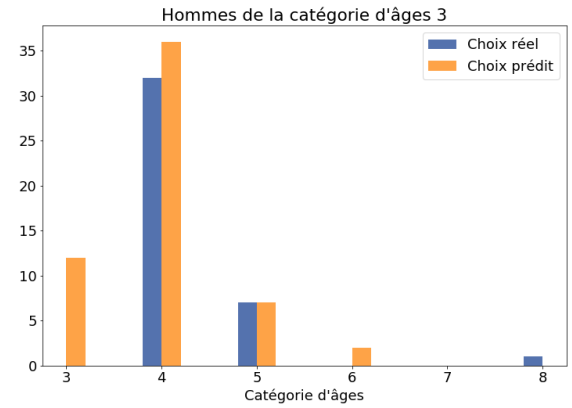
FIGURE 4.8 – Boite à moustaches de catégorie d'âges choisie par les hommes pour le réseau de neurones

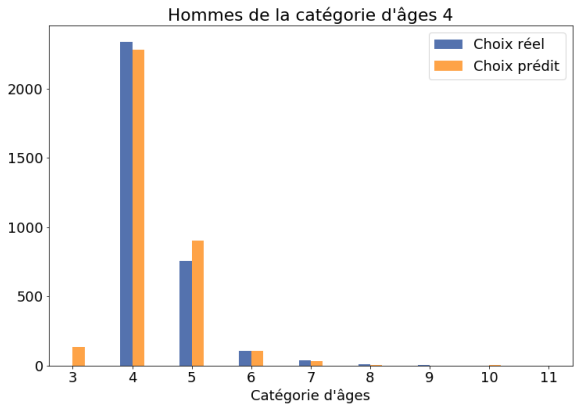
gramme, qui expliquent une valeur d'indicateur plus élevée. Toutefois, l'allure des deux distributions correspond. Cette analyse n'est pas suffisante pour affirmer que le modèle est une bonne représentation de la réalité. Il faut encore que ces choix soient effectués de manière similaire à nos données de test.

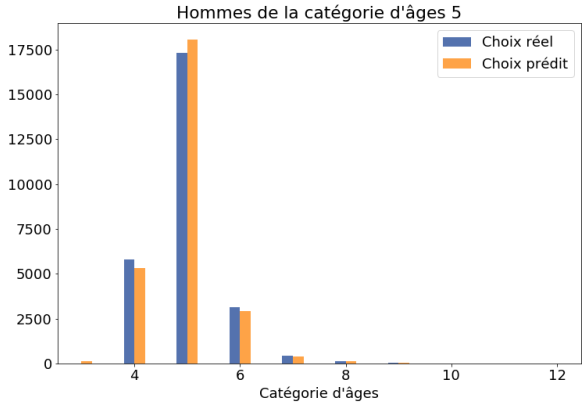
Nous avons alors calculé, pour chaque catégorie d'hommes, la distribution de leur préférence. Les histogrammes les contenant sont disponibles à la figure 4.9. Ces visualisations sont importantes puisqu'elles permettent d'émettre des conclusions déterminantes pour la suite. L'allure des distributions des prédictions correspond mieux à la réalité que les précédentes, à la figure 4.3. Les différences remarquables sont situées chez les hommes âgés, peu nombreux dans la population. Ces graphiques contiennent plusieurs choix improbables, non relevés précédemment. Pour exemple, selon notre modèle, un homme de catégorie 17, soit âgé de 85 à 90 ans, possède une préférence pour les femmes âgées de 30 à 35 ans.

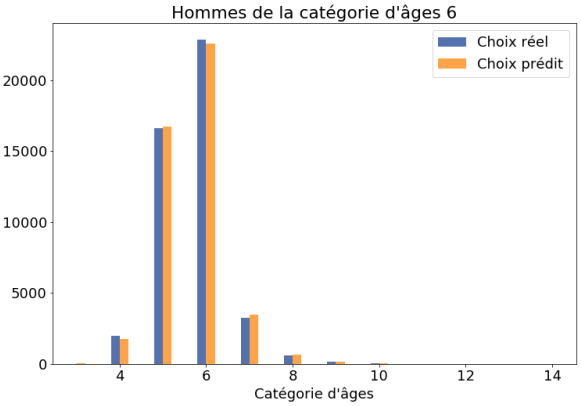
Nous avons ensuite étudié les relations entre la catégorie d'âges de la femme préférée et celle de l'homme, en réalité ainsi que selon la prédiction. Les deux graphes, sont observables à la figure 4.10. Les deux nuages de points sont semblables avec une majorité des individus située dans les classes 4 à 7. Tant en réalité, que dans les prédictions, nous relevons des choix extrêmes. Il existe un homme âgé de 35 à 40 ans qui a épousé une femme de catégorie 17. Notre modèle, quant à lui, prédit qu'au moins un homme de 85 à 90 ans désire marier une femme âgée de 30 à 35 ans. Ce constat est relevé à la visualisation précédente.

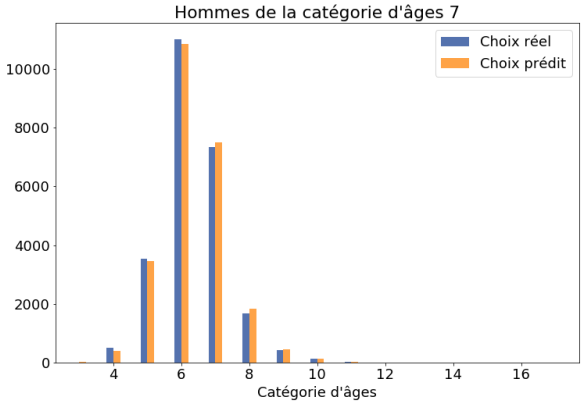
Terminons cette section par une dernière analyse : la distribution des différences d'âges entre l'homme et sa préférence. L'histogramme correspondant est situé à la figure 4.11. Les deux distributions ont un comportement similaire. Les différences remarquées sont faibles en comparaison de celles relevées sur le graphique 4.5. Grâce à ces analyses, nous pouvons conclure à la section suivante sur notre choix pour la suite de ce chapitre.

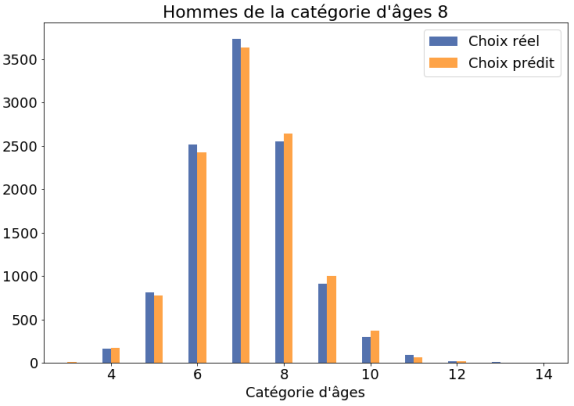

Hommes de la catégorie d'âges 3
Choix réel
Choix prédit
Catégorie d'âges

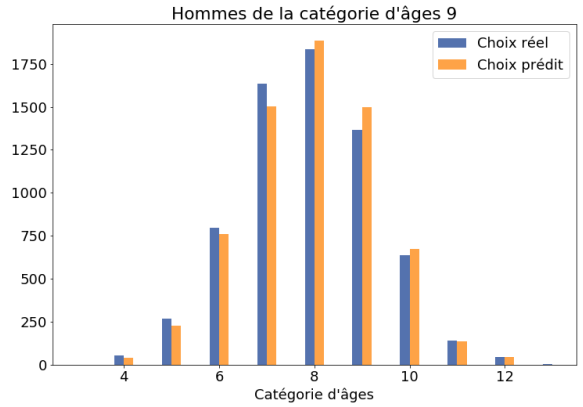

Hommes de la catégorie d'âges 4
Choix réel
Choix prédit
Catégorie d'âges

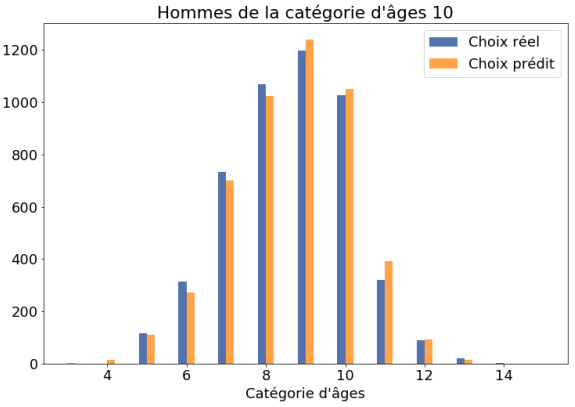

Hommes de la catégorie d'âges 5
Choix réel
Choix prédit
Catégorie d'âges


Hommes de la catégorie d'âges 6
Choix réel
Choix prédit
Catégorie d'âges


Hommes de la catégorie d'âges 7
Choix réel
Choix prédit
Catégorie d'âges


Hommes de la catégorie d'âges 8
Choix réel
Choix prédit
Catégorie d'âges


Hommes de la catégorie d'âges 9
Choix réel
Choix prédit
Catégorie d'âges


Hommes de la catégorie d'âges 10
Choix réel
Choix prédit
Catégorie d'âges

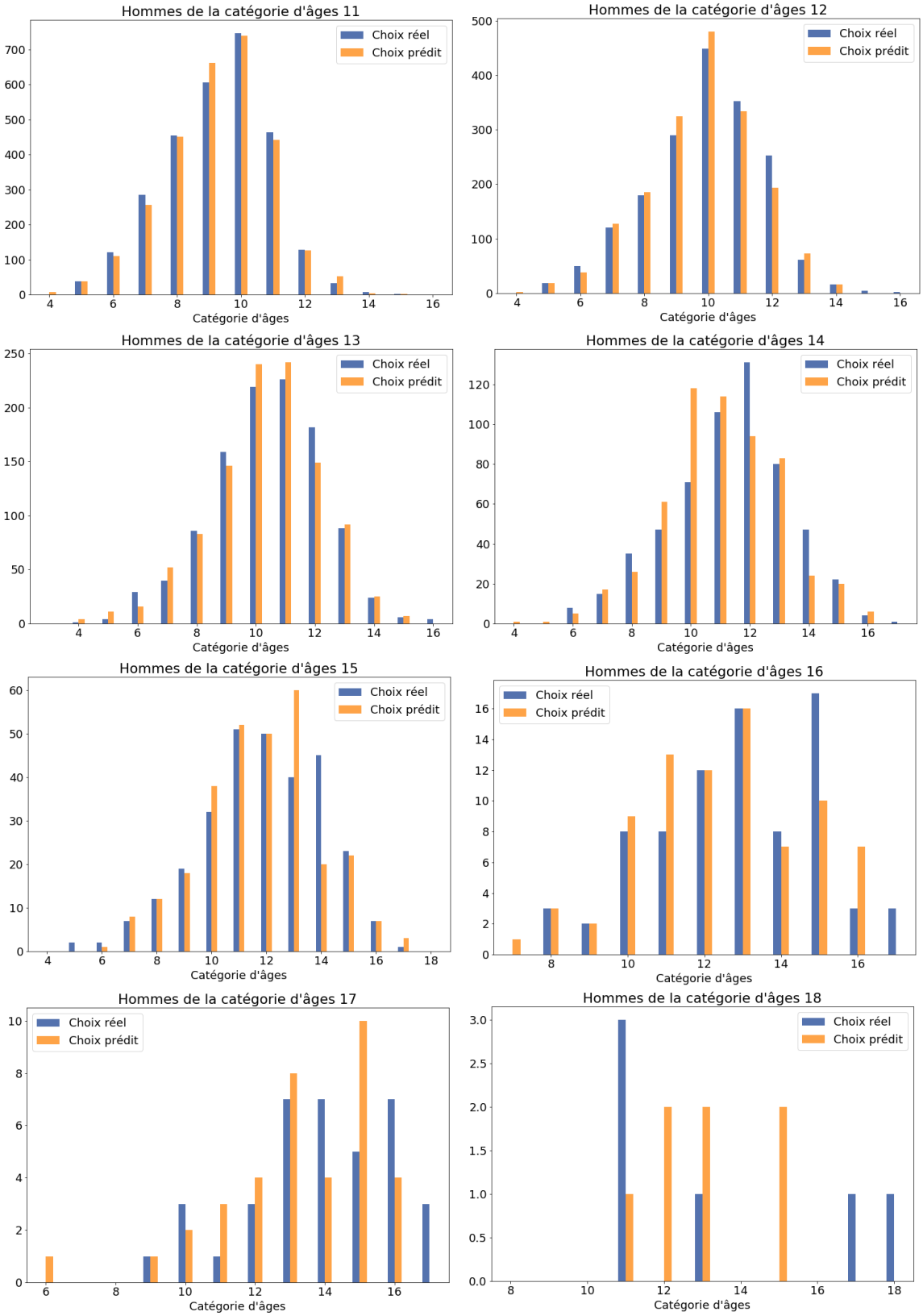
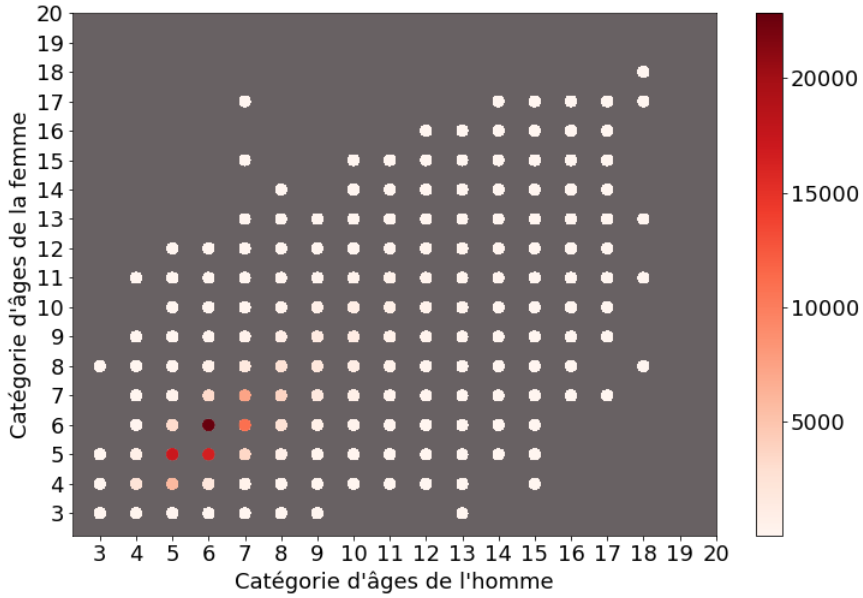
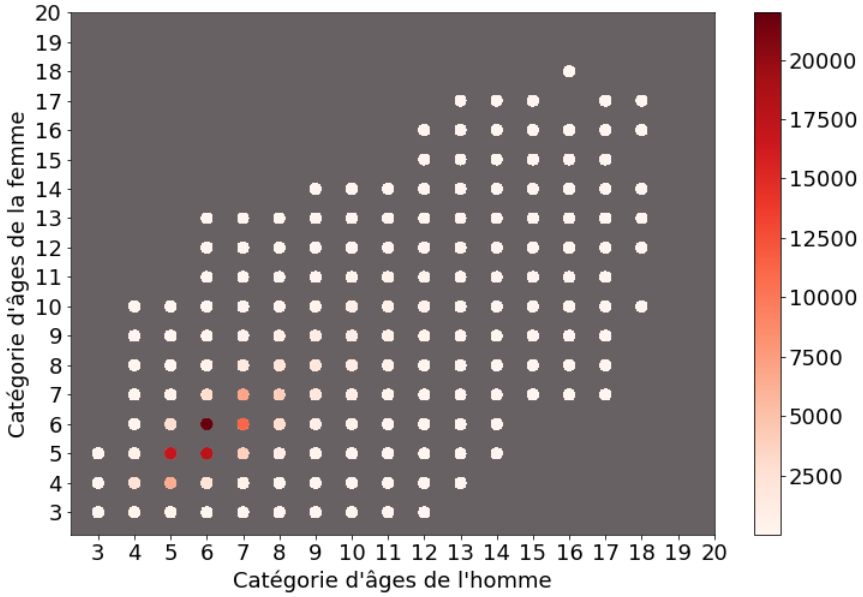


FIGURE 4.9 – Distribution des hommes selon la catégorie d'âges choisie par groupe d'âges masculin pour le réseau de neurones



(a) Relation entre la catégorie d'âges des époux réels



(b) Relation entre la catégorie d'âges de l'homme et sa prédiction

FIGURE 4.10 – Relation entre l'âge des hommes et leur préférence pour le réseau de neurones

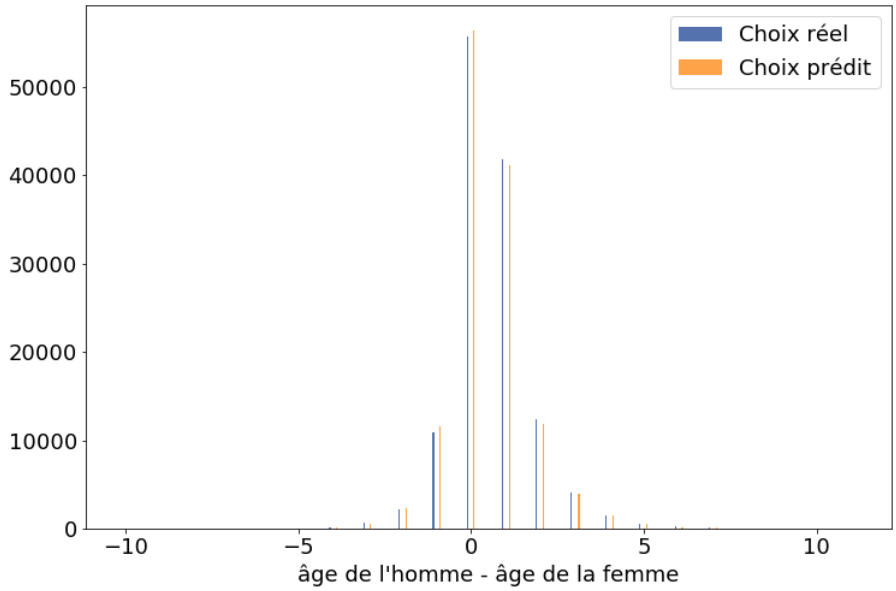


FIGURE 4.11 – Distribution des différences de catégorie d'âges entre l'homme et sa préférence pour le réseau de neurones

4.1.3 Modèle sélectionné

Nous concluons cette section en choisissant le modèle le plus adapté pour la suite de notre travail. La décision est moins évidente que celle prise à la section 3.3. Nous allons alors débattre et justifier notre intention de conserver le réseau de neurones, de nouveau.

Tout d'abord, tous deux ne pouvaient être comparés sur des indicateurs populaires, le ρ^2 et l'accuracy pour le premier et second modèle respectivement. En effet, leur valeur était faible mais justifiée. Nous devons alors confronter le modèle de choix et le réseau de neurones au niveau des graphes engendrés ainsi que sur des détails techniques. En étudiant uniquement les résultats graphiques, nous devons conserver le réseau de neurones puisqu'il retourne des distributions plus réalistes par catégorie d'âges d'hommes. De plus, la distribution des différences entre leur préférence et leur âge est proche de la distribution des écarts des âges des mariés réels. Ce constat n'est pas vérifié pour le modèle de choix. Toutefois, le réseau de neurones engendre quelques préférences peu probables mais en réalité, nous relevons des mariages extrêmes où la différence d'âges entre les époux est élevée. Gardons en mémoire que les résultats du modèle de choix discrets ne sont pas pour autant mauvais. Discutons alors du temps de résolution des deux algorithmes. Une fois calibré, le modèle de choix demande plus de ressources temporelles mais l'écart relevé est faible, environ une minute. Nous ne pouvons donc pas utiliser ce critère pour décider du modèle à conserver. Ces raisons nous poussent alors à préserver le deuxième modèle, soit le réseau de neurones, pour la suite de notre solution.

4.2 Formation des couples

Cette section est celle qui termine notre solution informatique à la problématique initiale : l'implémentation d'un algorithme simulant la dynamique des mariages belges. En effet, son objectif est de fournir une méthode réalisant le couplage des individus. Nous avons, précédemment, déterminé les préférences des hommes. Nous pouvons nous en servir pour leur associer une femme se trouvant dans le marché des mariages.

Deux méthodes de résolution sont proposées en commençant par une solution peu complexe, une méthode probabiliste. Celle-ci est déclinée avec plusieurs variantes pour offrir de meilleurs résultats. La seconde proposition est une méthode d'optimisation combinatoire, le recuit simulé. Avant de mentionner son implémentation et son analyse, nous expliquons la théorie derrière cet algorithme.

4.2.1 Méthodes probabilistes

Notre choix d'utiliser cette théorie, reposant sur le hasard pour obtenir des réponses est justifié par le peu de complexité qu'elle amène. De plus, les algorithmes probabilistes sont souvent très rapides tout en offrant des résultats exploitables. Nous débutons avec une première ébauche utilisant un unique tirage de nombres aléatoires. Nous le complexifions en affaiblissant les attentes des hommes envers leur conjointe idéale. Enfin, nous proposons une modification réalisant un couplage total des individus de la population.

Première ébauche

Commençons par détailler notre raisonnement pour ce premier modèle. Étant donné que nous possédons, pour chaque homme, la probabilité de choisir chaque femme - chaque catégorie d'âges - dans l'ensemble discret, nous avons pensé à implémenter une méthode assez simpliste reposant sur cette constatation.

Pour construire cette solution, nous avons d'abord assemblé les femmes du marché des mariages dans un tableau trié par classe d'âges. À chaque itération, nous considérons un homme provenant de ce marché. Nous observons son choix grâce aux préférences calculées auparavant et à un nombre aléatoire. Si, dans le tableau, il reste une femme dont l'âge correspond à ce choix, nous l'associons à cet homme et l'enlevons de notre ensemble. Si, au contraire, la catégorie d'âges attendue est dépourvue de femme, l'homme reste célibataire. L'implémentation de cette solution est disponible à l'annexe A.14.

Discutons alors de cette proposition. En couplant les hommes avec leur première préférence uniquement, nous n'épuisons pas tous les célibataires du marché. En effet, grâce aux analyses réalisées à la section 4.1, nous savons que les préférences déterminées par nos modèles ne correspondent pas parfaitement à la distribution des mariages réels. Nous ne pouvons donc pas obtenir un couplage total des individus dans notre marché pour cette méthode. Néanmoins, la finalité où des individus attendent un temps supplémentaire pour se marier est envisageable. En effet, le maintien de personnes une année additionnelle dans le marché des mariages est proposé dans la littérature, notamment dans le cadre du projet PRIMA. De plus, ce comportement humain est observable dans la réalité. Plusieurs individus désirent se marier au cours d'une année mais ne rencontrent pas de conjoint acceptable.

Dans le cas où cet algorithme est implanté dans une population synthétique, il faut envisager un traitement pour les individus n'ayant pas trouvé de partenaire. Deux solutions sont proposées. Tout d'abord, nous pouvons les laisser dans la population initiale sans réaliser de modification. Ces personnes, l'année suivante, doivent de nouveau passer par toutes les étapes de notre solution. Il est donc possible qu'elles ne soient plus dans le marché l'année subséquente et donc ne désirent plus se marier. Cette solution est facilement implémentable mais, pour nous, ne modélise pas correctement la vie réelle. La deuxième solution est plus complexe mais convient mieux. Il suffit de retenir les individus n'ayant pas trouvé de compagnon - par leur ID, par exemple - et de les placer en haut de la liste l'année d'après. Ils sont alors les premiers à chercher un partenaire et ont donc plus de chance de trouver la personne désirée. Cette dernière proposition modélise mieux, selon nous, le comportement réel de la population. Quand une personne désire se marier une année particulière mais qu'elle ne trouve pas de partenaire éligible, elle continue ses recherches l'année consécutive.

Effectuons quelques analyses sur cet algorithme afin d'étudier sa qualité. Tout d'abord, nous avons calculé le nombre réel de mariages et celui prédit sur nos données de test après avoir réalisé le couplage des individus. Nous obtenons $129\,219$ couples, en moyenne sur dix simulations, au lieu de $130\,260$, soit $99,2\%$ du nombre réel. Cela signifie que 1041 mariages n'ont pas été formés. Rappelons que ces informations concernent les couples dont le mariage s'est produit entre 1991 et 2000.

Il est alors intéressant d'étudier les caractéristiques des personnes n'ayant pas trouvé de partenaire idéal dans le marché des mariages. Nous avons affiché les distributions des individus, pour les hommes et pour les femmes, au niveau des catégories d'âges. Les deux histogrammes sont observables à la figure 4.12. Nous remarquons une forte présence d'hommes âgés de 30 à 40 ans alors que les femmes restant seules sont plus âgées. Il est important de remarquer la présence d'hommes dans beaucoup de catégories d'âges. Ceci est causé par notre modèle homme-dominant. Il peut être alors judicieux de trouver une astuce pour augmenter le nombre de mariages en couplant certaines femmes avec ces hommes de tous âges. Ceci nous mène au modèle, explicité dans la section subséquente.

Après avoir analysé les individus restant célibataires, nous voulons étudier les mariages produits. Il n'est pas intéressant de réaliser des visualisations sur ce sujet puisque nous observerions des résultats similaires aux figures 4.10 et 4.11. En effet, étant donné que nous considérons uniquement le premier choix des hommes et que nous effectuons une seule itération, nous engendrons des graphes semblables. Néanmoins, ils ne sont pas identiques aux précédents puisque quelques individus ne se sont pas mariés et que l'aléatoire entre en compte. L'analyse reste tout de même analogue.

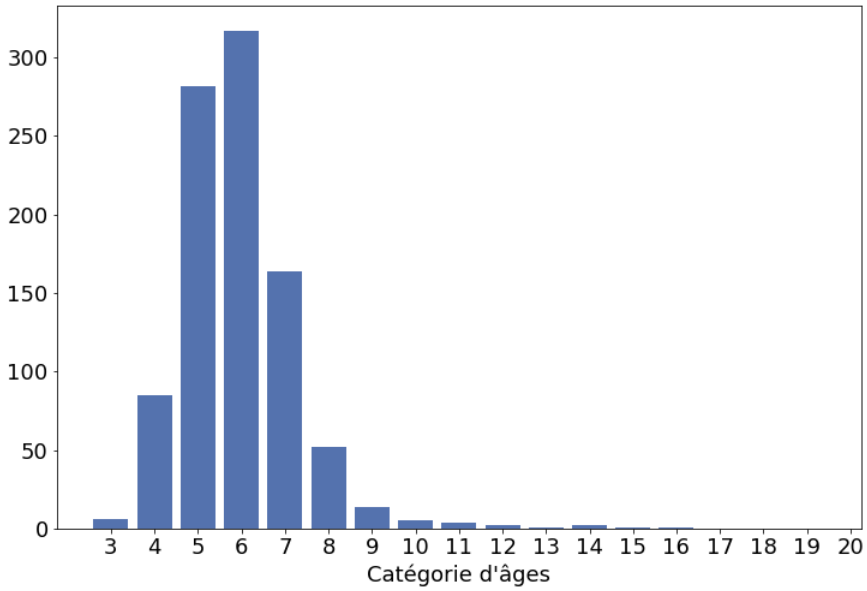
Nous avons relevé deux points positifs de ce modèle. Tout d'abord, il demande peu de ressources temporelles. Une fois les préférences calculées, son exécution se déroule en 24 secondes, durée moyenne sur dix simulations. Ensuite, il est robuste pour des nombres divergents d'hommes et femmes au sein du marché des mariages. Cette dernière constatation est non négligeable puisque c'est une situation commune. Si, toutefois, la quête du couplage total est recherchée, ce modèle n'est pas envisageable.

Première complexification

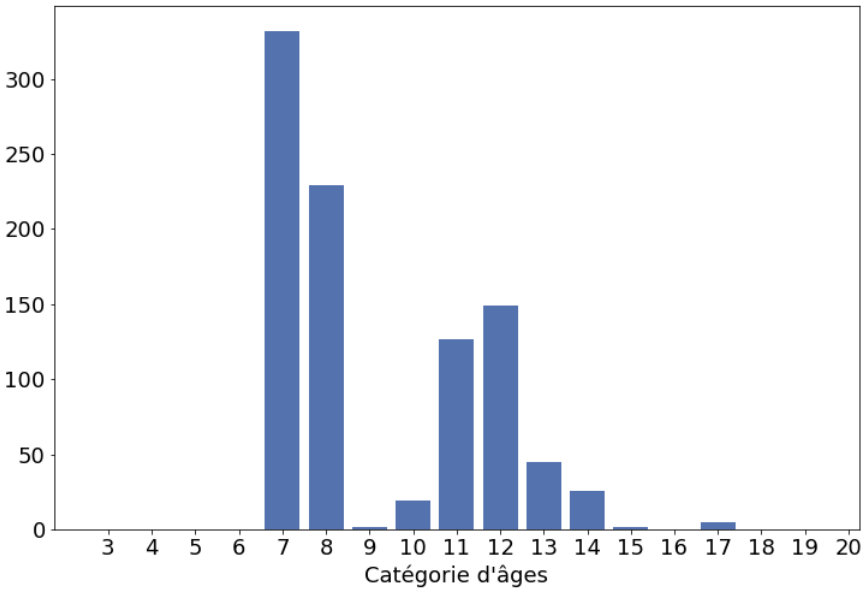
La deuxième solution proposée utilise notre première ébauche et tente alors d'améliorer les problèmes rencontrés. Cette deuxième idée essaye d'augmenter le nombre de mariages formés dans le marché en simulant un comportement proche de la réalité : relâcher ses attentes au niveau du partenaire utopique. Si un homme ne trouve pas de femme idéale dans le marché des mariages, il décide d'élargir ses préférences. Au point de vue algorithmique, cela se traduit comme suit : si un homme ne trouve pas de femme dans la catégorie d'âges préférée, il cherche alors dans une classe d'âges attenante à celle-ci. Nous remarquons qu'un homme a tendance à choisir une femme d'âge proche mais inférieur au sien. Cette constatation provient notamment de la distribution réelle des différences d'âges, présente à la figure 4.11. Si la recherche s'est soldée par un échec, nous décidons, en conséquence, de chercher dans la catégorie d'âges inférieure au choix.

Nous avons commencé par afficher le nombre moyen de mariages prédits sur dix simulations, soit $129\,339$. Cela représente $99,3\%$ de $130\,260$, le total réel. Nous avons engendré 120 mariages supplémentaires par rapport au premier modèle. Notons que ce gain ne demande pas de ressource temporelle additionnelle puisqu'une itération s'exécute en 25 secondes, moyenne sur dix simulations. C'est une seconde de plus que le premier modèle.

Nous possédons tout de même des craintes envers ce modèle ; il est possible que les distributions de mariages ne respectent plus la réalité puisque les individus ne choisissent pas forcément leur femme idéale.



(a) Résultat concernant les hommes



(b) Résultat concernant les femmes

FIGURE 4.12 – Distribution des personnes restant célibataires par catégorie d'âges pour la première version du modèle probabiliste

Vérifions cela grâce à une analyse graphique. Nous commençons par afficher les caractéristiques des célibataires restant dans le marché du mariage, tant pour les femmes que pour les hommes. Les deux histogrammes sont disponibles à la figure 4.13. Nous observons que certaines catégories d'âges chez les femmes restent peuplées, comme la 5, la 9 ou encore la 12. Alors que pour les hommes, il subsiste des individus dans toutes les catégories de 3 à 15. Ce constat indique la non-optimalité de notre modèle. En effet, suite à l'analyse de l'histogramme (a), nous pouvons dire que les femmes ont encore le choix dans beaucoup de catégories. Cette constatation est due à la caractéristique homme-dominant de notre modèle. Les femmes ne possèdent pas de préférence et ne peuvent donc pas choisir de partenaire. Si nous optons pour un modèle femme-dominant, nous retrouvons cette analyse mais pour les hommes. Ce modèle amène une faible amélioration par rapport à notre première ébauche.

Il est toutefois intéressant de conserver cette proposition plutôt que la première. En effet, les mariages engendrés sont proches statistiquement de la réalité puisque les liens entre les époux sont semblables à ceux présents au graphe 4.10 et que la distribution des différences d'âges au sein des couples est similaire à celle de la figure 4.11. Cela s'explique par la faible quantité d'hommes ayant relâché leur contrainte sur la femme idéale.

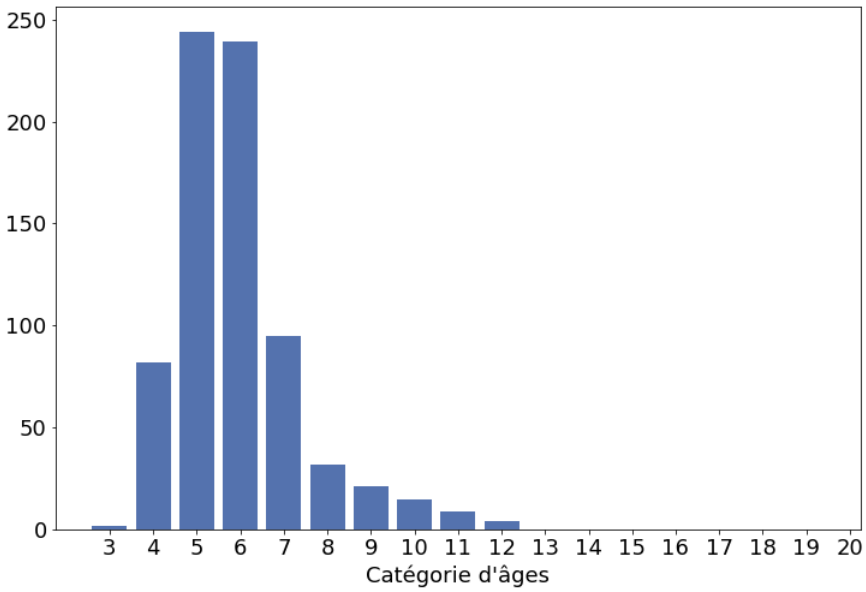
Concluons sur ce dernier modèle. Nous observons une faible évolution par rapport au premier algorithme avec un nombre prédit de mariages légèrement supérieur. La distribution des couples formés suite à cette modification reste semblable à la réalité. Cette solution est toujours applicable sur un marché ne contenant pas une équité d'hommes et de femmes. Néanmoins, le couplage n'est pas encore optimal puisqu'il subsiste des individus dans le marché des mariages et notamment des hommes aux âges variés. Pour palier à ce problème, nous avons engendré une dernière solution qui réalise un couplage total.

Méthode finale probabiliste

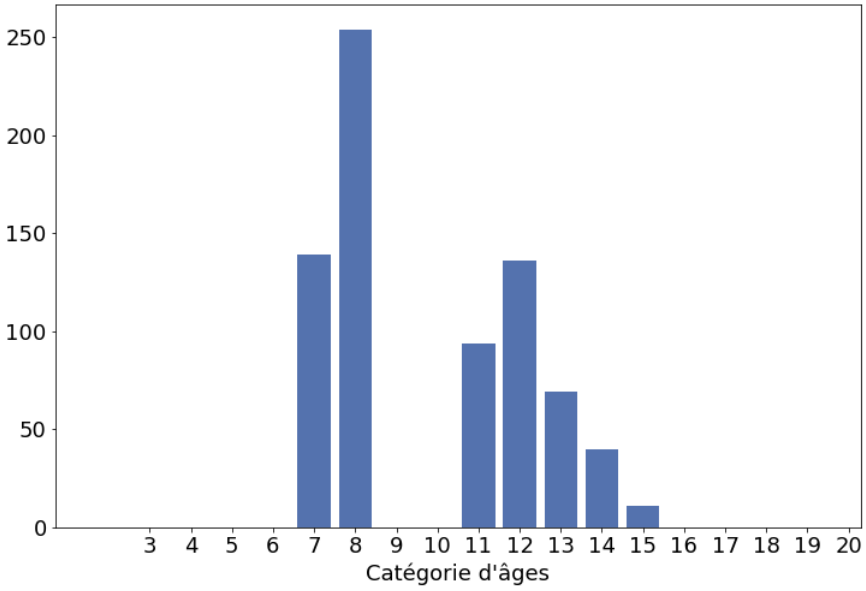
Ce dernier modèle proposé utilise notre première idée pour la complexifier et tenter d'obtenir un nombre de couples plus important, voire parfait. La première itération de l'algorithme est similaire au modèle précédent ; les hommes tentent de se marier avec une femme de leur groupe d'âges préféré. Si ils trouvent une conjointe répondant à ce critère, ils se marient et sinon ils restent célibataires. Nous gardons ensuite en mémoire les hommes n'ayant pas trouvé de partenaire pour recommencer ce processus en définissant un nouveau choix grâce à un second tirage de nombres aléatoires. Dans un premier temps, le modèle effectue cette opération dix fois. Ce chiffre est choisi suite à l'étude de la littérature réalisée auparavant. Le projet MOEBIUS [COR2015] annonce qu'un célibataire rencontre dix personnes par an. C'est ce que nous avons modélisé dans cette méthode. En effet, si un individu ne trouve pas la femme idéale à la première itération, il lui reste neuf essais pour y arriver.

Nous avons pratiqué la même étude que pour la première version présentée, en discutant, en premier lieu du nombre de couples engendrés, soit $129\,855$ en moyenne pour dix simulations. Cela représente $99,7\%$ de $130\,260$. Moins de 500 hommes subsistent dans notre marché des mariages. C'est une amélioration par rapport à la première modification de l'ébauche probabiliste. Cette complexification demande, néanmoins, plus de ressources temporelles. Elle s'exécute, en moyenne, en 1060 secondes ou 17 minutes et 40 secondes.

Étant donné la présence d'individus dans le marché des mariages après ces dix itérations, il est intéressant d'analyser leur caractéristique. Cela permet d'émettre des hypothèses sur leur échec à trouver un conjoint. Nous avons affiché deux histogrammes à la figure 4.14, un pour les hommes et le second pour les femmes. Commençons par l'analyse du graphe (a) : nous notons, de nouveau, la présence d'individus dans plusieurs catégories d'âges. Ce modèle n'est toujours pas optimal puisque les femmes

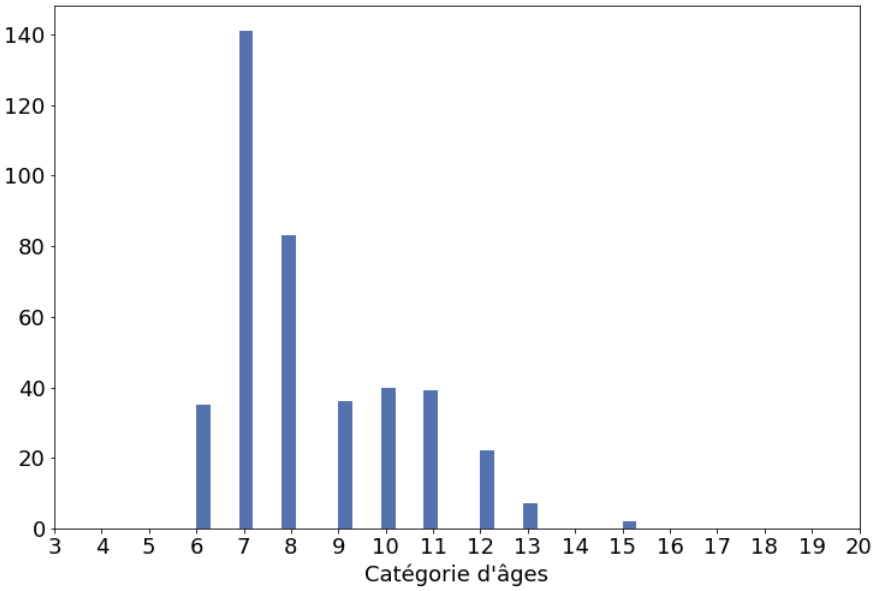


(a) Résultat concernant les hommes

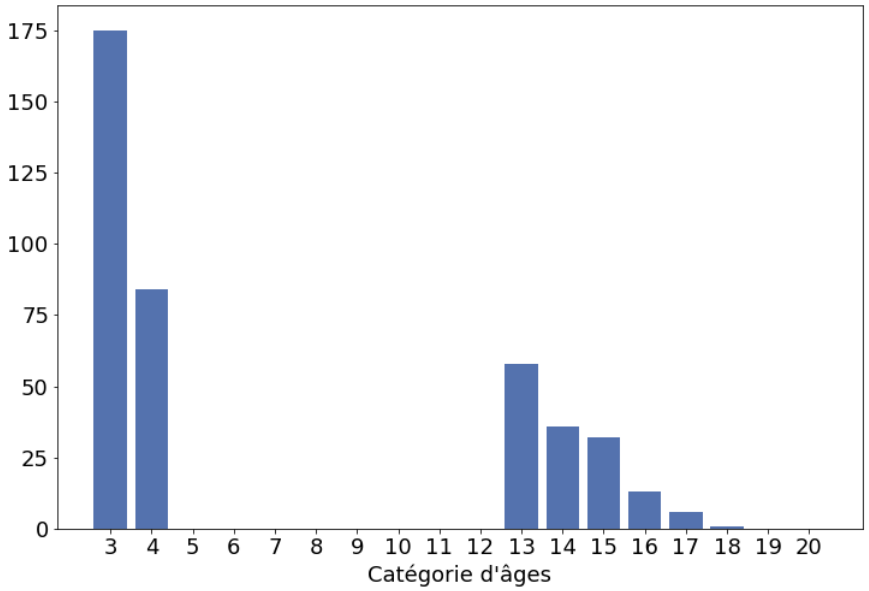


(b) Résultat concernant les femmes

FIGURE 4.13 – Distribution des personnes restant célibataires par catégorie d'âges pour la deuxième version du modèle probabiliste



(a) Résultat concernant les hommes



(b) Résultat concernant les femmes

FIGURE 4.14 – Distribution des personnes restant célibataires par catégorie d'âges pour la troisième version du modèle probabiliste

n'ayant pas été choisies pourraient trouver un partenaire idéal dû à la diversité des hommes présents. Le marché contient beaucoup d'individus masculins âgés de 35 à 40 ans alors que c'est la catégorie 3 qui est la plus présente pour les femmes. Ces dernières sont moins diversifiées. Les classes d'âges 5 à 12 chez les femmes ont été épuisées lors de la résolution de cette solution. Il est donc à présent difficile de penser obtenir un couplage total entre les individus n'ayant pas trouvé d'époux. En effet, il est moins probable d'obtenir des couples cohérents avec ces deux distributions. Cette constatation provient des analyses précédentes où, dans nos données, les différences d'âges entre les conjoints sont minimes. Notons tout de même que les mariages formés respectent les caractéristiques réelles des couples. Nous ne détaillons pas davantage ce constat car les graphes engendrés ont des allures similaires aux figures 4.10 et 4.11.

Nous avons enfin affiché, à la figure 4.15, un graphe du nombre de couples formés en fonction des itérations afin d'étudier la progression des mariages. Nous observons une fonction strictement croissante à tendance logarithmique. Celle-ci tend à se rapprocher de sa limite, le nombre réel de couples dans les données. La pente de cette fonction décroit assez rapidement, après quelques itérations.

Ce modèle, bien qu'utilisant plus de ressources temporelles, retourne un nombre plus élevé de mariages. Nous nous approchons du couplage total des individus mais plusieurs centaines de personnes sont toujours présentes dans le marché. Deux conclusions divergentes surviennent ; soit il est peut-être intéressant de s'arrêter à 4 ou 5 itérations puisqu'après la progression est minime ou soit continuer ces itérations jusqu'à atteindre la limite si le temps d'exécution est relativement peu élevé. Nous pouvons alors décider d'augmenter le nombre d'itérations pour engendrer de meilleures performances. Notons que le coût du temps d'exécution pour ajouter quelques itérations est minime car en avançant dans celles-ci, le nombre d'hommes présents dans le marché diminue fortement. Néanmoins, l'analyse précédente nous indique que ce nombre peut ne pas augmenter, mais plutôt stagner, et donc ne jamais atteindre un couplage total.

Nous décidons de réaliser cette troisième proposition avec non plus 10 mais 500 itérations. Cette exécution est réalisée en 20 minutes et 40 secondes, soit 3 minutes de plus que l'accomplissement de dix

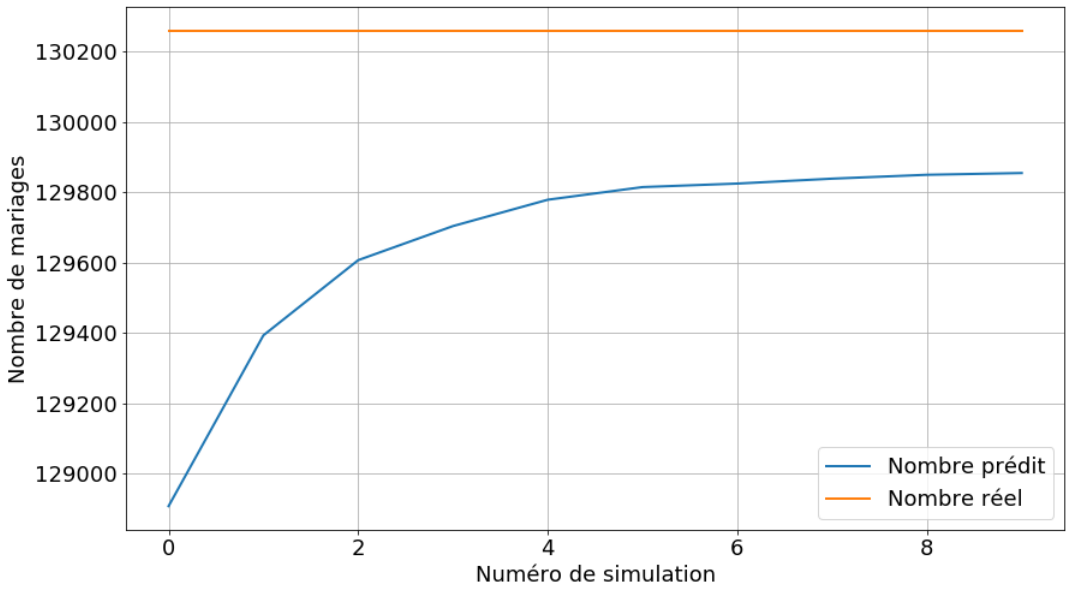


FIGURE 4.15 – Nombre de mariages prédits à chaque simulation pour la troisième version du modèle probabiliste

simulations. Nous voulons alors analyser la croissance du nombre de mariages. Nous avons affiché ces informations dans le graphique 4.16. Nous observons une convergence lente vers le nombre réel de mariages. Passer de 10 à 500 itérations demande peu de ressources temporelles. Il est donc peut-être intéressant d'augmenter ce nombre pour approcher le matching parfait, soit un couplage de tous les individus présents dans le marché des mariages. C'est ce que nous avons effectué.

Nous avons modifié quelque peu l'algorithme pour que celui s'arrête quand un couplage parfait est trouvé. Après 171 784 itérations, le marché des mariages est vide. Nous avons affiché, à la figure 4.17, l'évolution du total de mariages par itération. Après quelques milliers d'itérations, la progression de cette quantité est faible, voire nulle. Plusieurs itérations ne prédisent pas de mariage ou un seul. Il faut être prudent lors de l'utilisation du nombre d'itérations nécessaire. Travaillant sur une base aléatoire, nous observons de grandes variations dans la convergence. L'étude de cette fluctuation peut être intéressante mais demande des ressources temporelles importantes. En effet, cette méthode possède un temps d'exécution plus grand que lors de nos dix simulations, 5 heures 12 minutes 58 secondes. L'origine de ce temps plus important vient de la convergence lente de notre méthode. Beaucoup d'itérations après 25 000 ne prévoient aucun mariage mais parcourent tout de même les individus restant dans le marché. Lancer plusieurs dizaines de simulations n'était pas envisageable. Nous possédons quelques appréhensions envers cette méthode. Ayant forcé notre marché à se vider, nous pensons que des partenaires aux caractéristiques dissemblables sont mariés. Si ces époux sont relativement peu nombreux, nous nous trouvons en situation réelle où quelques exceptions sont remarquées. C'est pourquoi nous réalisons une courte analyse des mariages formés.

Nous avons alors étudié cette composition de couples en observant la relation réelle et celle prédite du groupe d'âges des deux partenaires. Deux nuages de points rassemblant ces informations sont disponibles à la figure 4.18. Les deux graphes ont des allures assez similaires. Nous remarquons tout de même quelques mariages extrêmes dans les prédictions. Plusieurs jeunes femmes sont couplées avec des hommes assez âgés. Néanmoins ceux-ci étant peu nombreux, les résultats obtenus semblent cohérents.

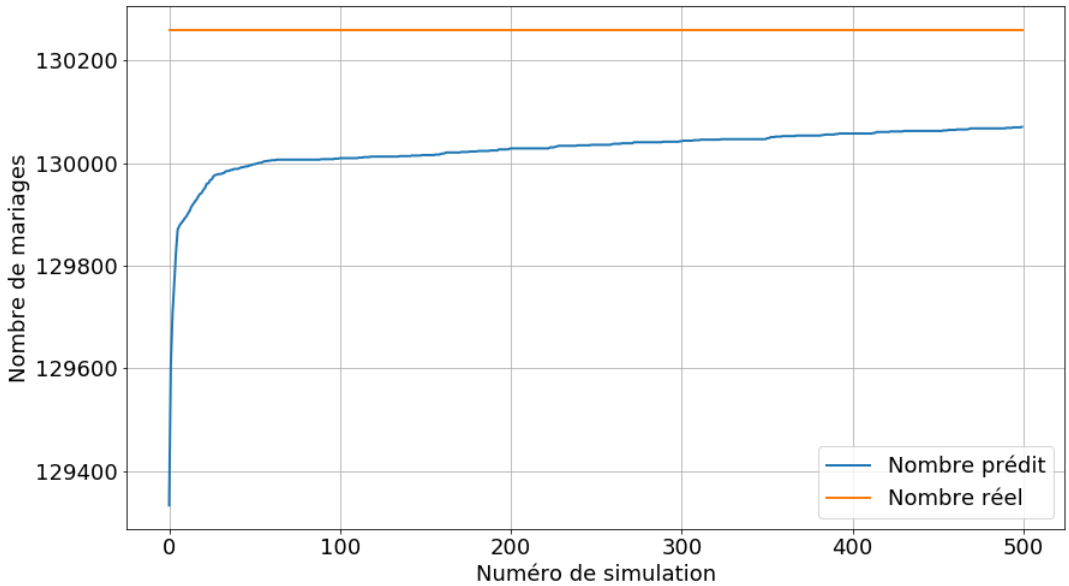


FIGURE 4.16 – Nombre de mariages prédits à chaque simulation pour la troisième version du modèle probabiliste avec 500 itérations

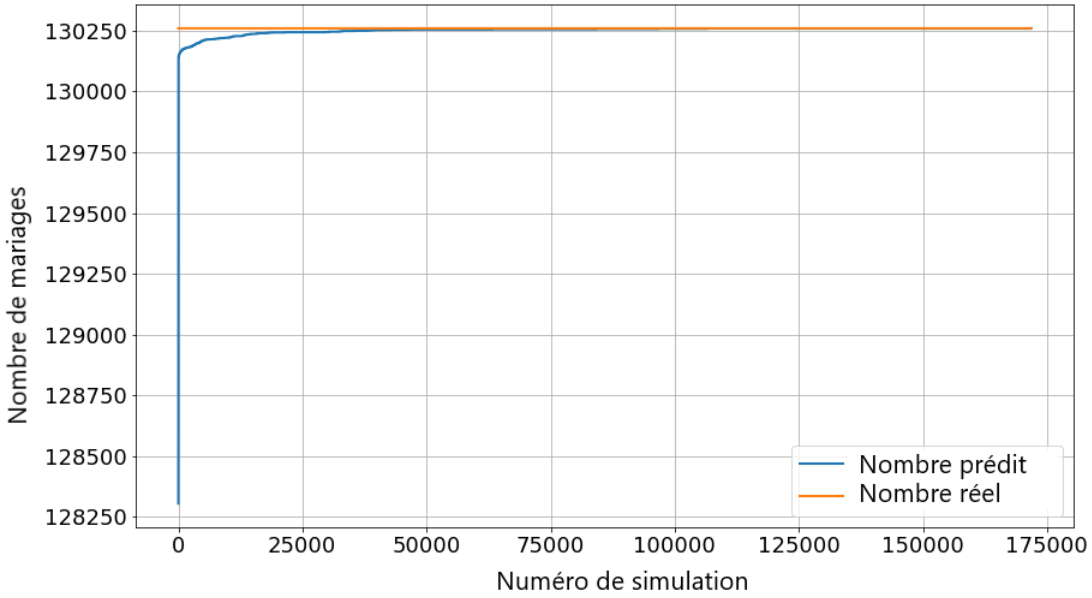


FIGURE 4.17 – Nombre de mariages prédits à chaque simulation pour le modèle probabiliste avec couplage total

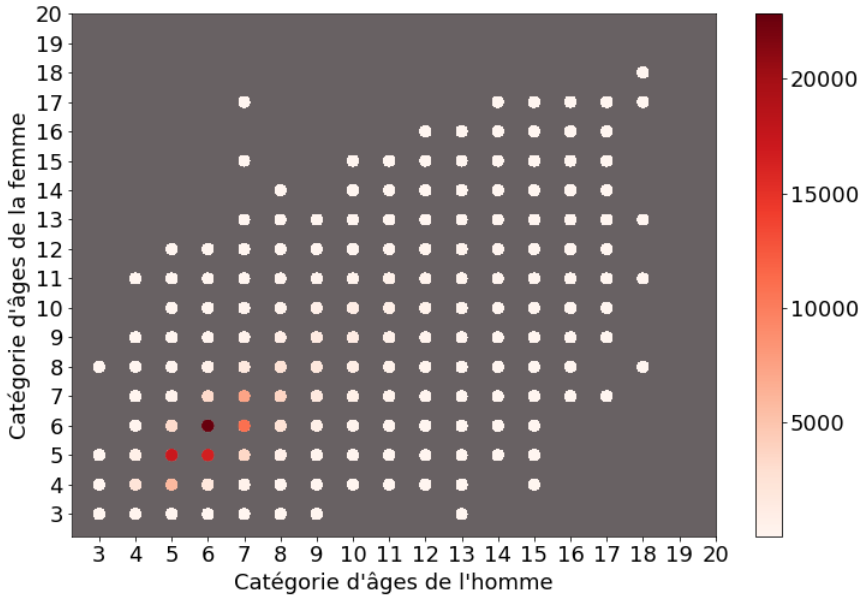
Terminons par l'analyse de la distribution des différences d'âges au sein des couples, à la figure 4.19. Nous obtenons deux distributions à l'apparence semblable. Beaucoup de couples possèdent la même catégorie d'âges ou l'homme, une classe supérieure à celle de son épouse. Quelques différences sont observées mais les mariages formés semblent assez proches de la réalité. Nous n'observons pas ou peu, dans les prédictions, des écarts extrêmes entre l'âge des partenaires.

Cette dernière méthode doit être adaptée si le marché des mariages contient un nombre d'hommes différent de celui des femmes puisqu'un couplage total ne peut être réalisé. La solution que nous proposons est le retrait aléatoire des individus du sexe le plus présent jusqu'à obtenir une équité dans le marché. Ces personnes doivent donc attendre une année supplémentaire pour se marier.

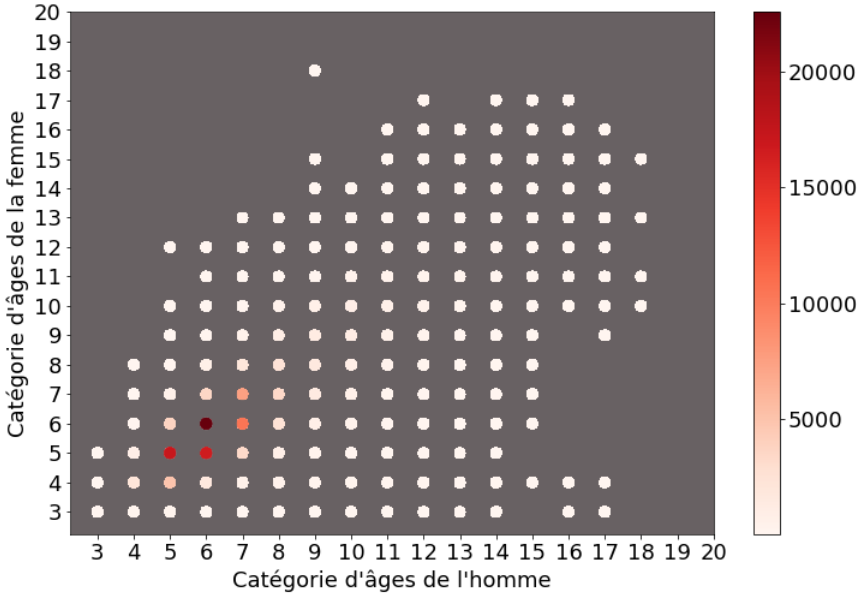
Concluons cette section en justifiant notre choix de conserver une seule proposition de modèles : celle offrant un couplage total des individus. Elle demande plus de ressources que les autres modèles étudiés mais les résultats obtenus sont plus intéressants. En effet, ce modèle couple chaque individu du marché des mariages en respectant assez bien les caractéristiques des mariages réels. De plus, avec quelques opérations simples, il est possible de gérer le problème de la présence non équilibrée des hommes et des femmes dans le marché.

4.2.2 Méthode du recuit simulé

Nous terminons donc cette section par une dernière méthode provenant de la littérature pour effectuer le couplage des individus du marché des mariages : celle du recuit simulé. Celle-ci offre un couplage total des individus mais en contre-partie amène un risque de mariages extrêmes si les méta-paramètres la composant ne sont pas appropriés. Nous avons décidé d'explorer cette méthode d'une part, pour sa facilité d'implémentation et d'autre part, pour ses propriétés de convergence intéressantes. Nous commençons cette section par une description théorique de l'algorithme grâce aux informations des ressources [BEN2009] et [ELL2017]. Ensuite, nous expliquons son implémentation en Python pour terminer avec son analyse.



(a) Relation entre la catégorie d'âges des époux réels



(b) Relation entre la catégorie d'âges des époux prédits

FIGURE 4.18 – Relation entre l'âge des conjoints pour le modèle probabiliste avec couplage total

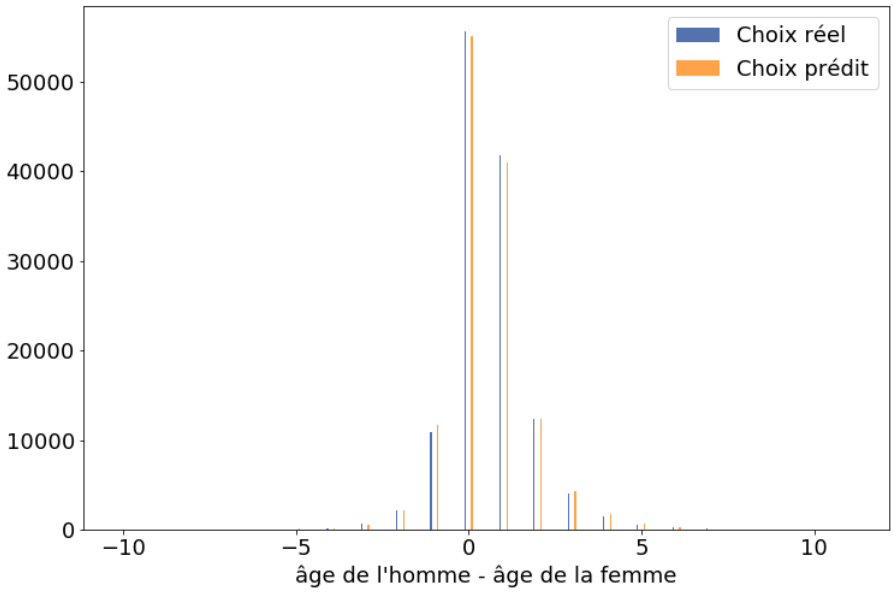


FIGURE 4.19 – Distribution des différences de catégorie d'âges au sein des mariages pour le modèle probabiliste avec couplage total

Théorie

La méthode du recuit simulé (simulated annealing en anglais) est un algorithme empirique, inventé par trois chercheurs d'IBM en 1983 [1] pour résoudre des problèmes d'optimisation combinatoire. Elle imite un processus populaire en métallurgie qui permet d'améliorer la qualité d'un métal solide. Détaillons cette méthode métallurgique. Au cours de ce processus, les chercheurs tentent d'atteindre un état d'énergie minimal correspondant à une structure stable du métal. Ils commencent à travailler sous haute température avec le métal liquide. Ensuite, débute la phase de refroidissement où la matière reprend forme solide en diminuant progressivement la température.

Détaillons maintenant la méthode implémentée en 1983. Nous disposons d'un ensemble fini d'éléments, E, et d'une fonction à minimiser, $H(x)$, où $x \in E$. Nous désirons trouver x tel que $H(x)$ soit minimal. Comme E est fini, nous pourrions le parcourir en entier afin de trouver le minimum global de la fonction H en calculant pour chaque x, $H(x)$. Cette solution est irréalisable si le cardinal de E est élevé. Une seconde idée est de parcourir E élément par élément en utilisant une notion de voisinage. Nous passons d'un élément x vers son voisin, y, si $H(y) \leq H(x)$. La méthode s'arrête quand $H(x) < H(y) \ \ \forall y \in E$ où y est un voisin de x. Nous relevons deux défauts majeurs à cette méthode : H doit être calculé de nombreuses fois et il est fort probable de se trouver coincé dans un minimum local. Pour contrer ces défauts, les chercheurs d'IBM ont inventé le recuit simulé.

Le parcours de l'ensemble E pour la méthode du recuit simulé tient compte d'une notion d'aléatoire. À l'instant t, nous nous trouvons à l'élément x. Nous choisissons ensuite un voisin y aléatoirement. Si $H(y) \leq H(x)$, nous effectuons un mouvement vers y au temps $t+1$. Au contraire, si $H(y) > H(x)$, nous n'excluons pas la possibilité de migrer vers y. En effet, la probabilité de se déplacer en y vaut $e^{-\frac{H(y)-H(x)}{T}}$ où T est appelé température et est un réel positif. Cette solution permet d'éviter les minima locaux qui ne sont pas globaux.

1. Kirkpatrick, S., Gelatt, C., Vecchi, M., *Optimization by Simulated Annealing*, Science, New Series, 220 (4598), pages 671-680, 1983.

Formalisons ce développement avec quelques détails supplémentaires et une comparaison avec le domaine de la métallurgie. Comme explicité précédemment, nous effectuons des mouvements entre les éléments de E selon une distribution de probabilité dépendant de la qualité de leurs voisins. Les voisins donnant des valeurs plus basses de la fonction objectif, H, ont plus de chance d'être parcourus. Cette probabilité dépend d'un paramètre, T, la température. En métallurgie, celle-ci occupe aussi une place importante dans le processus. Pour nous aider dans la suite de nos explications, un pseudo-code de cette méthode est disponible ci-après.

Fonction Recuit_simulé **:**

Choisir aléatoirement un élément, x_0, initial, $x = x_0$

Définir la température initiale, $T = T_0$

while « condition à définir » **:**

nbr_moves = 0

for i allant de 1 à iter_palier **:**

Calculer un voisin, y, de x
Calculer $\Delta = H(y) - H(x)$

if CritMetropolis(Δ, T) **:**
$x = y$
nbr_moves += 1

acceptance_rate = nbr_moves/iter_palier
Mise à jour de T : $T = T * coef$

Fonction CritMetropolis(Δ, T) **:**

if $\Delta \leq 0$, **return TRUE**

else avec une probabilité $e^{\frac{-\Delta}{T}}$, **return TRUE**

else return FALSE

La fonction CritMetropolis est stochastique et retourne l'élément suivant dans notre processus, soit l'élément actuel - nous n'effectuons pas de mouvement -, soit son voisin. Notons qu'un voisin qui améliore la fonction objectif ou dont la valeur est égale à celle de l'élément actuel est toujours visité.

La température, T, doit maintenant être définie. Plusieurs possibilités peuvent être envisagées. En effet, la température est un paramètre à ne pas négliger car elle contrôle l'acceptation des voisins n'apportant aucune amélioration. Si celle-ci est élevée, les voisins possèdent tous la même probabilité d'être visités. Si, au contraire, elle est faible, les mouvements engendrant des fortes hausses dans la fonction de coût possèdent une faible probabilité d'être parcourus. Enfin, si elle est nulle, aucune élévation de la

fonction H n'est acceptée. La température peut être fixe mais la faire varier en fonction des itérations est conseillé. Il est intéressant de choisir un T grand pour débuter et de le faire tendre vers 0 au cours des itérations.

La fonction qui définit l'évolution de la température est le schéma de refroidissement. Ce nom provient du processus de métallurgie dont le recuit simulé est tiré. Si la température reste constante, la méthode porte le nom d'algorithme de Metropolis. Le paramètre T peut décroitre de manière géométrique ou arithmétique à chaque itération. Le schéma de refroidissement utilisé dans notre mémoire est une décroissance par palier. Plusieurs itérations sont effectuées avant d'engendrer une mise à jour de la température. Nous choisissons de réaliser cette baisse de manière géométrique. Enfin, des schémas plus complexes peuvent être utilisés avec notamment des remontées de température.

Terminons cette section théorique en discutant des réglages des méta-paramètres. Nous devons déterminer la température initiale, le critère d'arrêt ou encore fixer le schéma de refroidissement. C'est un rôle assez compliqué. Nous possédons seulement quelques pistes dans la littérature. La température initiale ne doit pas être trop grande sinon les premiers mouvements seront inutiles. Pour le critère d'arrêt, nous pouvons fixer un nombre d'itérations maximum mais une autre idée est à explorer. Nous pouvons calculer la proportion d'acceptation, définie par le quotient entre le nombre réel de mouvements lors d'un palier et le nombre d'itérations que ce dernier comprend. Si ce pourcentage est faible, il est inutile de continuer la recherche.

Implémentation et analyse

Pour appliquer la théorie du recuit simulé sur notre problématique, plusieurs éléments doivent être définis. Détaillons alors l'implémentation de cette solution effectuant un couplage total des individus présents dans le marché des mariages. Notre notebook Jupyter prévu à cet effet est disponible à l'annexe A.15.

L'ensemble E contient toutes les possibilités de couplages des célibataires deux à deux. Étudier toutes les alternatives est donc inimaginable. Dans ce mémoire, nous ne tentons pas de minimiser une fonction mais de la maximiser. Celle-ci est définie comme suit ; à chaque individu masculin est associé un ensemble de probabilités, provenant de la section 4.1. Nous connaissons donc leurs préférences envers chaque femme du marché. Pour quantifier la qualité d'un couplage, nous observons la probabilité associée à chaque couple, soit la préférence de l'homme envers la femme qui lui est assignée. Nous voulons alors maximiser la somme de ces probabilités afin que chaque homme soit satisfait de son couple. Quelques changements minimes dans l'algorithme doivent être encourus pour tenir compte de la maximisation. Un élément voisin d'un autre de notre ensemble E reçoit cette dénomination si leur couplage est similaire à l'exception de deux couples dont les partenaires ont été interchangés. Par conséquent, pour calculer un voisin du matching actuel, il suffit de tirer aléatoirement deux hommes et d'échanger leur épouse.

Analysons à présent l'algorithme implémenté. Nous avons exécuté une première fois la méthode avec les paramètres suivants :

- température initiale : 15,
- itération maximale : $50\,000$,
- nombre d'itérations par palier : 50,
- borne du taux d'acceptation : $0,2$,

- coefficient de décroissance géométrique de la température : $0,9$.

Ils ont été choisis en étudiant la littérature mais aussi de manière arbitraire. En effet, nous savons que les méta-paramètres ne peuvent être fixés manuellement mais nous voulions obtenir de premiers résultats sans les travailler. Si de bonnes performances sont observées, nous tenterons de les modifier pour approcher les mariages réels de manière rapide et précise.

La méthode est assez lente mais converge en 6240 itérations. Son temps d'exécution vaut 12 heures 37 minutes 8 secondes. Le nombre d'itérations maximal n'étant pas atteint, l'arrêt de l'algorithme est dû à une valeur trop faible du taux d'acceptation. Beaucoup de couplages voisins n'amélioraient pas la fonction objectif et la température est devenue trop faible pour les accepter.

Pour étudier cette méthode, nous avons, en premier lieu, affiché la croissance de notre fonction objectif par itération. Le graphique est observable à la figure 4.20. Bien que cela n'est pas distinguable, cette fonction n'est pas strictement croissante. En effet, à certaines itérations, la valeur de la fonction objectif chute à cause de la fonction de Metropolis. Nous pouvons voir que globalement la fonction croit mais de manière assez lente. Il est important de préciser que la valeur de la fonction objectif pour les mariages réels n'est pas une limite à atteindre. En effet, étant donné la présence d'aléatoire lors du calcul des préférences, il est possible d'obtenir un couplage dont la valeur de cette fonction est plus élevée que celle du couplage réel.

Nous avons ensuite étudié les couples formés. Deux graphes sont construits, un nuage de points contenant les liens entre les époux et un histogramme affichant la distribution des différences d'âges au sein des mariages créés. Ils sont respectivement disponibles aux figures 4.21 et 4.22. Les nuages de points n'ont pas même allure. Celui attaché au recuit simulé contient une masse de points indiquant la présence de mariages extrêmes. Nous ne distinguons pas de schéma dans ce graphe. La visualisation affichant les relations réelles des mariés montre que les points semblent s'agglomérer autour de la droite identité. Les deux graphes se ressemblent uniquement au niveau de la présence importante d'individus âgés de 25 à 40 ans et de la quantité abondante de mariages dont les époux appartiennent à la classe d'âges 6. Ces constatations sont aussi observables dans l'histogramme. La formation de couples dont la différence d'âges est élevée est remarquée.

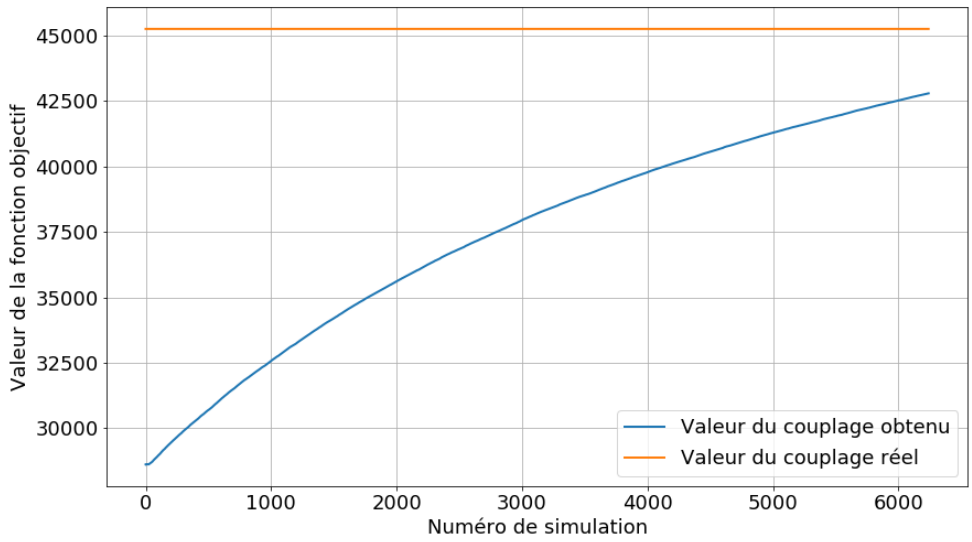
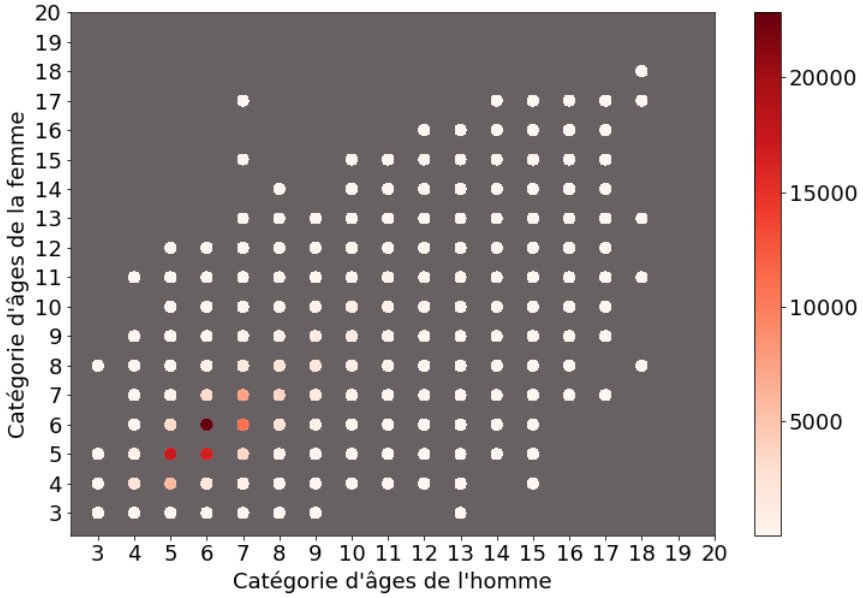
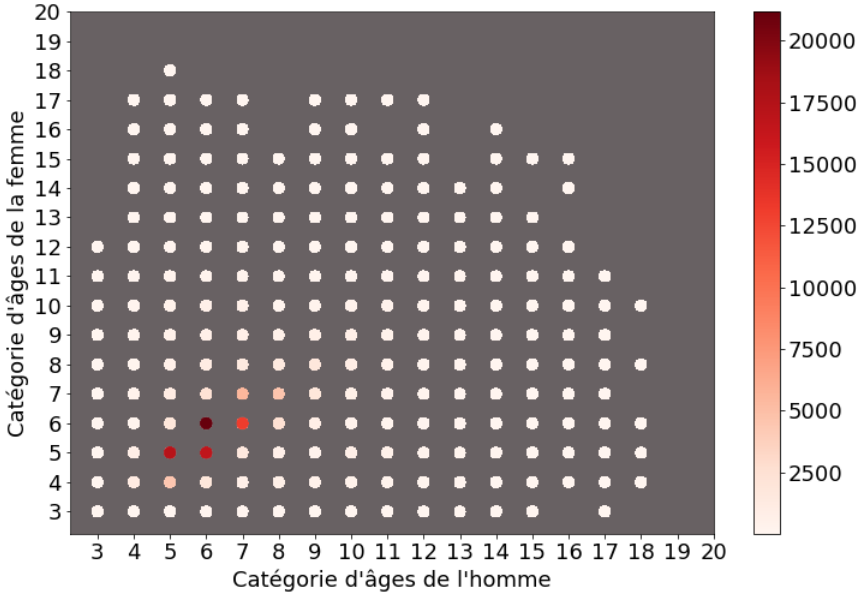


FIGURE 4.20 – Évolution de la fonction objectif par itération pour le recuit simulé



(a) Relation entre la catégorie d'âges des époux réels



(b) Relation entre la catégorie d'âges des époux prédits

FIGURE 4.21 – Relation entre l'âge des conjoints pour le recuit simulé

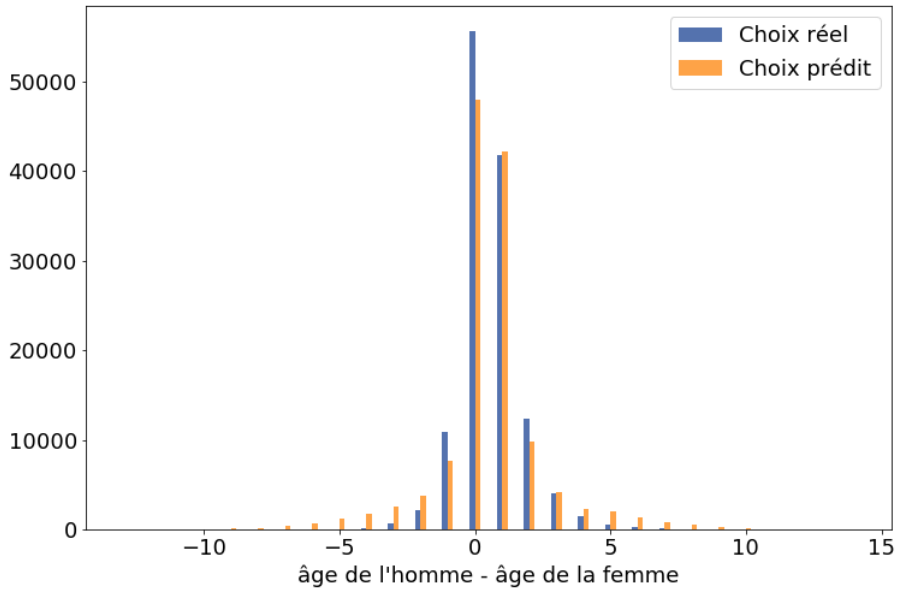


FIGURE 4.22 – Distribution des différences de catégorie d'âges au sein des mariages pour le recuit simulé

De nouveau, pour pouvoir intégrer cette solution dans un marché contenant des nombres différents d'individus masculins et féminins, des modifications doivent être encourues. La solution proposée est celle détaillée à la section précédente : le retrait aléatoire d'individus pour obtenir une équité parfaite dans le marché des mariages.

Concluons, à présent, sur cette méthode. Elle est très lente puisqu'elle s'exécute en une dizaine d'heures. Avec d'autres méta-paramètres, notamment d'autres valeurs de température, ce temps pourrait peut-être s'améliorer. Néanmoins, cet algorithme ne sera jamais rapide car à chaque itération, nous échangeons uniquement les partenaires de deux hommes. Nos données étant relativement nombreuses et débutant avec un couplage aléatoire, de multiples itérations sont alors nécessaires. Ensuite, les mariages obtenus ne modélisent pas correctement la réalité puisque plusieurs couples aux âges divergents ont été formés. Ce problème peut être amélioré en diminuant la borne du taux d'acceptation puisque le nombre d'itérations requis va croitre. Cette solution amène néanmoins un temps de résolution plus important. Nous avons testé cette possibilité et bien que les mariages obtenus soient plus cohérents avec la réalité, nous conservons des couples extrêmes. Pour n'en posséder que quelques uns, il faudrait uniquement considérer la condition de fin de boucle sur les itérations et indiquer un nombre maximal important. Cette solution n'est pas envisageable puisque sa résolution demanderait trop de ressources temporelles. Ces explications indiquent que d'autres tests peuvent être effectués sur les méta-paramètres. Néanmoins, selon nous, cette méthode est trop lente pour être utilisée en situation réelle où le module des mariages est effectué chaque année sur plusieurs dizaines d'années. De plus, nous dénombrons cinq méta-paramètres ayant tous de l'influence sur ce modèle. Il n'est donc pas possible de réaliser une étude sur différentes combinaisons de valeurs de ceux-ci. Cela demanderait trop de ressources temporelles.

4.2.3 Comparaison des méthodes implémentées

Nous voulons à présent choisir la méthode à adopter, dans celles présentées précédemment, pour coupler les individus d'un marché des mariages. Tout d'abord, la première version de la méthode probabiliste est assez simpliste et ne demande pas beaucoup de ressources temporelles. Bien qu'elle ne couple pas tous les individus du marché des mariages, une majorité des hommes a trouvé la femme idéale.

Nous possédons alors une distribution proche de la réalité au niveau des couplages réalisés. Néanmoins, nous ne la conservons pas puisque tous les individus n'ont pas trouvé de partenaires. De plus, les hommes ayant subi un échec au cours de cette recherche sont d'âges variés. De nouveaux couples peuvent alors être formés tout en respectant la distribution des caractéristiques des mariages réels. Nous ne nous étendons pas non plus sur la première modification de cette méthode car les conclusions sont similaires.

La méthode probabiliste engendrant un couplage total des individus est intéressante. En effet, en un temps d'exécution moyen, elle engendre des mariages proches de ceux réellement produits. Tous les hommes se sont mariés. Le modèle provoque tout de même quelques mariages extrêmes mais ceux-ci sont peu nombreux.

Enfin, la méthode du recuit simulé doit, pour nous, être écartée. Son temps de résolution est trop important pour qu'elle soit intégrée dans une population synthétique. De plus, elle engendre un nombre considérable de mariages extrêmes au sein du marché. Ce problème pourrait être géré par l'étude des méta-paramètres du modèle. Néanmoins, nous pensons qu'il est peu probable de trouver une combinaison de ceux-ci amenant des résultats satisfaisants tout en s'exécutant en un temps relativement court.

Nous voulons tout de même souligner que toutes ces méthodes s'appuient sur une base importante d'aléatoire. Une simulation peut donc amener des résultats différents d'une autre. Toutefois, nous pensons que ces écarts ne sont pas trop élevés. Terminons cette section par une ouverture. Bien que la théorie du recuit simulé ne soit pas adaptée pour notre situation, elle a déjà fait ses preuves pour d'autres problèmes d'optimisation. Il est alors peut-être intéressant d'étudier d'autres méthodes évolutionnistes, par exemple les algorithmes génétiques, qui s'exécuteraient plus rapidement et dont les résultats seraient acceptables. Un futur travail peut donc se pencher sur ce sujet. Nous voulions aussi analyser le problème des mariages stables qui réalise un matching total dans la population mais la littérature sur ce sujet constatait la présence importante de couples extrêmes. Nos algorithmes ne possédant pas cette limitation, nous avons préféré rejeter cette théorie. Enfin, nous pouvions réaliser une étude similaire à la nôtre en incluant le type de diplômes des individus mais de nombreux diplômes indéterminés étant relevés, nous n'avons pas trouvé d'intérêt à cette idée.

4.3 Couplage des individus du marché des mariages construit au chapitre 3

Nous terminons ce chapitre en appliquant la méthode réalisant les couplages, sélectionnée à la section précédente, sur une situation concrète. En effet, nous avons construit au chapitre 3 un marché des mariages. Nous pouvons alors implémenter notre solution sur ce marché afin d'accomplir le couplage des individus. Nous utilisons les données de test, prévues à cet effet.

Nous commençons par conserver uniquement les personnes désirant se marier, choix déterminé par le réseau de neurones à la section 3.2. Notre marché contient à présent $11\,148$ hommes et $10\,738$ femmes. Ces nombres n'étant pas identiques, nous décidons d'enlever aléatoirement 410 individus masculins. Notre marché comporte donc $21\,476$ personnes contre $22\,400$ en réalité. Notre estimation engendre environ 1000 individus de moins que prévu. Une cause de cette constatation est le retrait d'hommes pour obtenir une équité au sein du marché des mariages. Nous avons ensuite calculé les préférences des hommes désirant se marier. Suite à cela, le couplage peut débuter. Analysons alors les caractéristiques des couples formés.

Nous avons affiché les liens entre la catégorie d'âges des partenaires des mariages réels et ceux prédits

aux figures 4.23 et 4.24 respectivement. L'allure des deux graphiques se ressemble mais nous notons quelques différences au niveau des mariages extrêmes. Tout d'abord, les points des deux graphes semblent s'amasser autour de la droite identité. La majorité des couples possède un âge compris entre 20 et 35 ans. Dans les prédictions nous trouvons quelques mariages extrêmes où la femme est âgée de 60 à 80 ans et l'homme plus jeune, de catégorie 6 à 8. Ces couples ne sont pas présents dans les données.

Nous retrouvons ce constat dans les histogrammes aux figures 4.25 et 4.26. Le premier représente

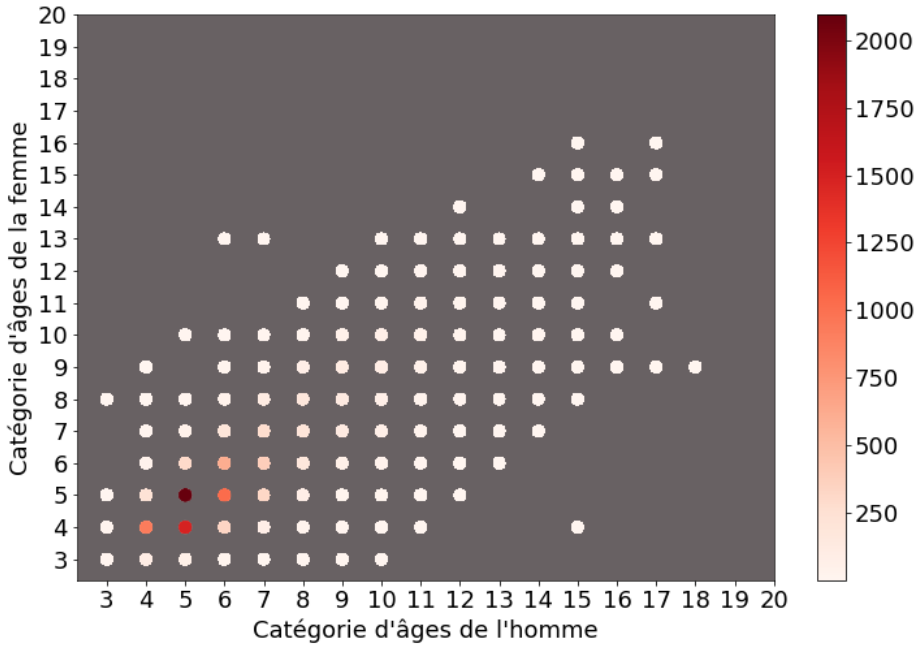


FIGURE 4.23 – Relation entre la catégorie d'âges des époux réels

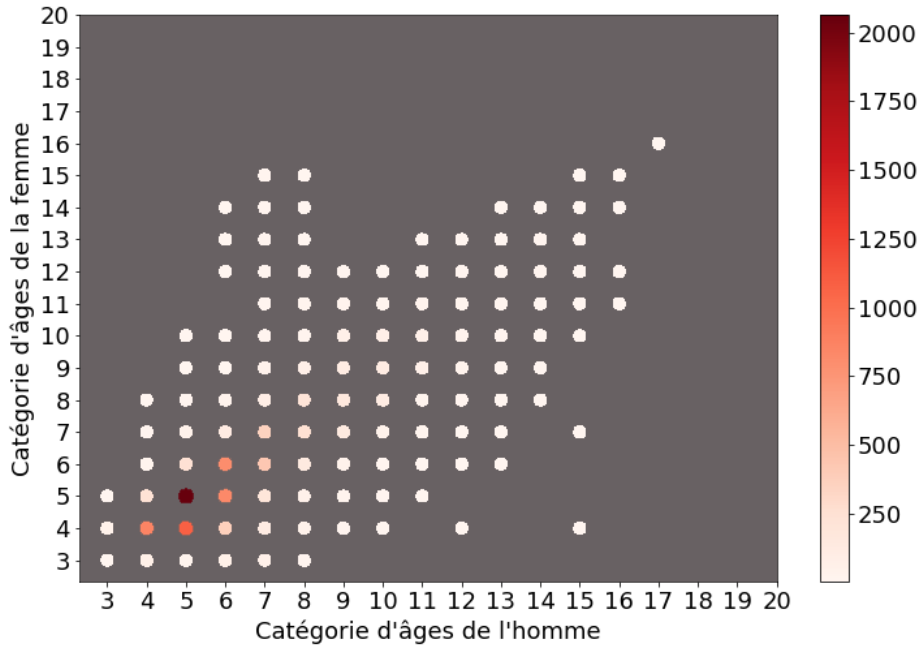


FIGURE 4.24 – Relation entre la catégorie d'âges des époux prédits

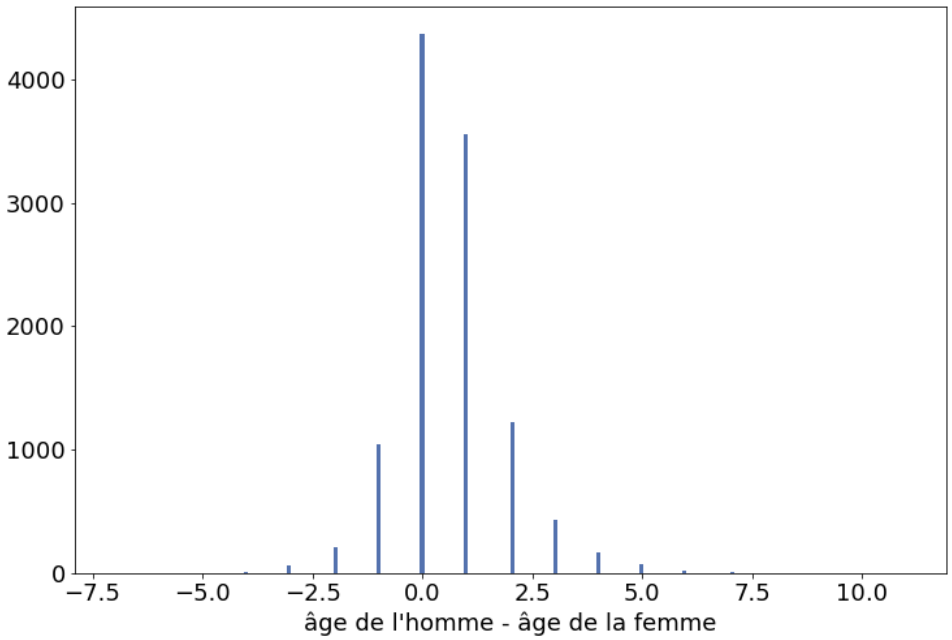


FIGURE 4.25 – Distribution réelle des différences de catégorie d'âges au sein des mariages

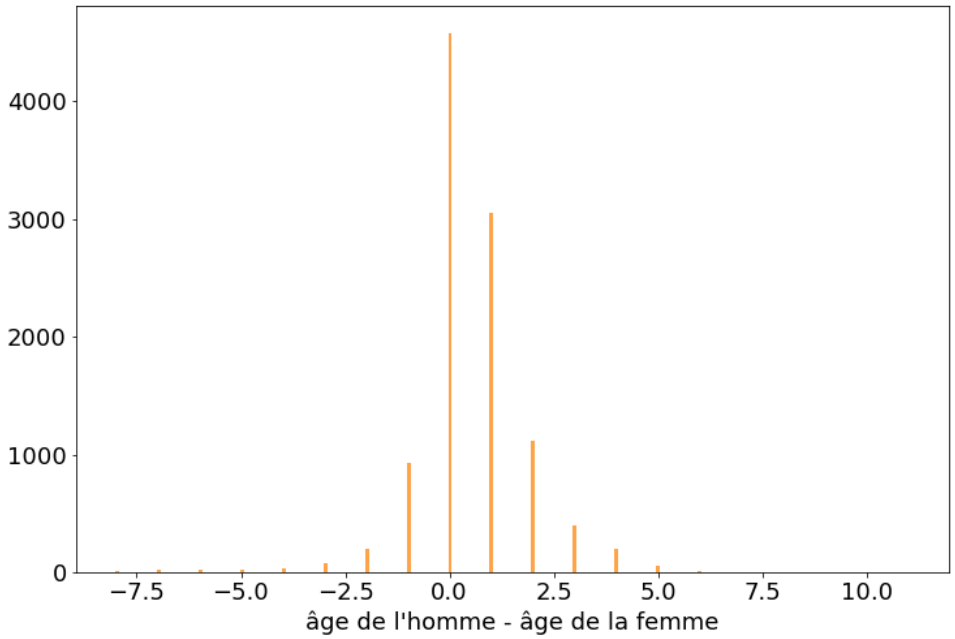


FIGURE 4.26 – Distribution prédite des différences de catégorie d'âges au sein des mariages

la distribution des différences d'âges au sein des couples s'étant mariés au cours de l'année 2000. Le second, quant à lui, affiche la distribution des mariages formés suite à l'application de notre solution. Ce dernier nous indique une faible présence de couples dont l'âge de la femme est fortement supérieur à celui de son conjoint. Or, ces mariages sont presque inexistants en réalité. Néanmoins, les deux histogrammes possèdent une allure similaire où une majeure partie des couples est pourvue d'un âge similaire ou d'une classe de différence avec un mari plus âgé. Ce dernier aspect où l'homme est plus vieux d'une catégorie est moins présent dans les prédictions que dans les données.

Nous pouvons à présent conclure sur la qualité de notre solution de couplage. Nous avons utilisé le marché des mariages construit au chapitre 3 qui tient compte des mariages produits au cours de l'année 2000. Ce chapitre, quant à lui, repose sur les données des mariages survenus entre 1991 et 2000. Cette vérification n'est donc pas conçue sur des données totalement inconnues. Néanmoins, nous pouvons affirmer certaines remarques concernant notre solution. Quelques mariages extrêmes, non présents dans les données, sont produits. Ils restent toutefois peu nombreux et dans la vie réelle, il est possible que des personnes aux âges différents se marient. De plus, de manière générale, les couples formés respectent assez bien la distribution réelle.

Chapitre 5

Implémentation sur une population virtuelle

Ce dernier chapitre tient plus d'une perspective que de la production de résultats. En effet, après l'obtention d'une solution qui prédit les mariages en Belgique, il est intéressant de l'implémenter au sein d'un projet de population synthétique. Nous avons développé la situation réelle de VBIH (VirtualBelgium In Health) puisque les données sur lesquelles le projet se repose sont similaires à celles que nous utilisons. Ce chapitre possède alors l'objectif d'appuyer l'utilité de nos résultats. Toutefois, ce travail exigeant beaucoup de ressources temporelles, nous n'avons pu tester ceux-ci sur la population de VBIH mais uniquement donner des pistes pour faciliter un futur travail.

Ce chapitre se divise alors en deux sections principales. La première vise à expliquer le projet VirtualBelgium In Health avec notamment le contexte de sa création, la génération de la population virtuelle et son évolution dynamique. Nous abordons évidemment en détail le module du mariage. Cette section est principalement écrite grâce aux ressources [VIL2013, NOL2013, NAX2016]. La seconde détaille les changements à encourir pour pouvoir implémenter nos résultats dans ce projet. Nous devons nous conformer au travail réalisé pour intégrer nos algorithmes.

Sommaire

5.1 VirtualBelgium in Health

Dans la société actuelle, deux phénomènes, menant au questionnement des autorités, se répandent : le vieillissement de la population et l'augmentation du nombre de personnes âgées vivant seules. L'objectif de la Wallonie est par conséquent d'assurer, malgré l'augmentation de la longévité de ses habitants, leur maintien en bonne santé et leur survie dans des conditions de vie décentes. Pour atteindre ce but, il faut planifier au mieux l'offre en santé publique et notamment en termes de soins ambulatoires qui sont en partie causés par la perte d'autonomie des aînés.

C'est dans ce contexte qu'est né VirtualBelgium in Health. La Région Wallonne a besoin d'un outil, destiné aux pouvoirs locaux et aux acteurs de terrain, permettant de prédire l'évolution des besoins en soins de santé. Les résultats doivent être exploitables au niveau régional mais aussi au niveau des communes wallonnes. Cela contribuera à la planification globale ainsi qu'à l'amélioration des politiques communales spécifiques puisqu'il y aura une meilleure connaissance locale des besoins futurs des personnes âgées. En effet, un des intérêts principaux de ce projet est l'amélioration de la qualité de vie de cette population. De plus, ce programme s'inscrit dans le domaine de la santé et de l'amélioration de la qualité de vie en tenant compte du développement durable. Les résultats doivent aussi être présentés de manière à ce qu'ils soient clairs, lisibles et soignés. Finalement, nous pouvons résumer en disant que VirtualBelgium in Health, c'est la création d'un outil de programmation pour les soins et services de santé qui s'appuie sur une estimation des besoins à une échelle spatiale fine, celle des communes.

Le projet VBIH, *VirtualBelgium in Health : Une plateforme de population virtuelle comme outil d'aide à la décision en matière de prospective et de planification des besoins de soins de santé pour les personnes âgées*, est financé par la Wallonie dans le cadre du programme WB Health pour une durée de 36 mois. Celle-ci a lancé un appel d'offres en 2013 pour ce projet ambitieux. L'Université de Namur (UNamur) a répondu à cette demande. Elle a donc pris le poste de promoteur et coordinateur du projet avec son unité de recherche, le Centre Namurois des Systèmes Complexes (NaXys) dont le représentant est Dr Eric Cornelis. L'Université Catholique de Louvain (UCL), par le biais de son centre de recherche en démographie, l'IACCHOS (l'Institut d'Analyse du Changement dans l'Histoire et les Sociétés Contemporaines), supplée ce projet. Pour compléter cette équipe, VBIH dispose d'un parrain : l'Observatoire Wallon de la Santé (l'OWS) qui, en échange de la disponibilité de l'outil développé et des résultats, s'engage à apporter son aide au projet. Il est sollicité pour fournir l'accès à certaines données nécessaires et pour ses connaissances dans le domaine de la santé publique.

L'enjeu de VirtualBelgium in Health est du domaine prospectif. Aujourd'hui, les prévisions liées aux soins de santé se basent sur les consommations actuelles et non pas sur l'évolution des besoins. Cet objectif sera atteint en utilisant la plateforme virtuelle pour qu'elle fournisse ces informations sur les personnes âgées. Une attention particulière est apportée aux besoins de type ambulatoire. Le déroulement du projet est donc le suivant : tout d'abord, une analyse des liens de corrélation entre les différentes caractéristiques de la population et leurs besoins en soins de santé a été réalisée. Ensuite, l'évolution temporelle de ces déterminants à une échelle géographique très fine est modélisée pour en retirer les perspectives. Pour la suite de cette section, il est important que nous précisions que nos informations proviennent du rapport de mi-parcours, non publié, datant du 1^{er} mars 2016 [NAX2016]. Certaines dynamiques ont pu, par conséquent, être améliorées depuis lors.

5.1.1 Génération de la population synthétique

Pour la génération de la population synthétique wallonne, les chercheurs de VBIH ont décidé d'implémenter une méthode robuste et adaptée à leurs données. Ces dernières ont d'ailleurs dû être récoltées

pour une année déterminée. Les chercheurs ont choisi 2011 comme année initiale puisqu'elle est proche de l'année de lancement du projet, 2013, tout en étant assez éloignée pour garantir la disponibilité des données statistiques. La méthode créée se déroule en deux étapes : tout d'abord, les individus sont « fabriqués » pour ensuite, les regrouper en ménages.

Afin de construire la population attendue, les chercheurs de ce projet ont dû respecter une contrainte : celle-ci doit contenir les individus groupés en ménages, localisés au sein des communes wallonnes. Afin que la population générée soit la plus proche de la réalité, elle doit être caractérisée par des attributs pertinents. Ceux-ci peuvent être propres aux individus, comme le diplôme, ou propres aux ménages, comme le type de ménage. Évidemment, les données désirées ne sont jamais toutes disponibles et ne possèdent pas souvent le même niveau d'agrégation. Certaines sont exploitables pour les communes, d'autres pour les régions ou encore pour les provinces. Cette problématique accentue la difficulté de créer une population synthétique proche statistiquement de la population réelle.

Pour générer tous les individus de la population, des données provenant du Registre National ainsi que du recensement administratif, Census2011 ont été nécessaires puisqu'elles donnent les informations sur les attributs de ces personnes. Les individus créés sont caractérisés par les variables *âge*, *genre*, *commune*, *taille* et *type du ménage*, *lien entre la personne et le chef de son ménage*. Les informations sur le diplôme et le statut professionnel ont été ajoutées par après. Voici une brève description de ces variables :

- *âge* : classes par tranche de 5 ans,
- *genre* : homme ou femme,
- *commune* : domicile de l'individu,
- *diplôme* : plus haut niveau obtenu dans la classification CITE, notation hiérarchique des types de diplômes, allant de 1 : « Enseignement primaire » à 8 : « Niveau doctorat ou équivalent »,
- *statut professionnel* : inactif, travailleur ou chômeur,
- *taille du ménage* : de « 1 » à « 6 et plus »,
- *type de ménage* : couples avec ou sans enfants, cohabitants avec ou sans enfants, familles monoparentales, ménages collectifs, personnes isolées ou autres,
- *lien avec le chef de ménage* : chef, conjoint, enfant, parent, ...

Une fois les individus créés, les chercheurs ont dû les assembler en ménages. Ils ont donc groupé les individus générés en sous-groupes de sorte que les ménages obtenus soient les plus proches statistiquement des ménages wallons existants. Pour ce faire, des statistiques sont disponibles, comme par exemple, celles indiquant les différences d'âge entre les conjoints d'un même couple.

Les chercheurs ont ensuite ajouté des caractéristiques relatives à la santé aux individus générés. L'accès aux données nécessaires a été facilité par le parrain du projet, l'OWS. Cette tâche s'est déroulée en trois étapes : tout d'abord, ils ont récolté les données statistiques disponibles liées à la santé. La deuxième partie concerne la modification de la population synthétique pour ajouter de nouvelles caractéristiques aux individus mais aussi aux ménages. Le dernier point porte sur l'ajout de particularités aux entités territoriales. Ces attributs sont, par exemple, le nombre de médecins, le nombre de lits en maison de repos, l'offre des soins à domicile ou encore la distance à l'hôpital le plus proche.

L'objectif de ce projet étant de prédire le besoin en soins de santé des personnes âgées, les chercheurs ont décidé d'ajouter des caractéristiques liées à la vieillesse. Pour ce faire, ils ont déterminé cinq groupes de maladies touchant la population âgée. Ces pathologies choisies peuvent toutes être identifiées par des médicaments qui leur sont spécifiques. Grâce aux données reçues, les personnes par âge, par genre et par commune, traitées par ces médicaments spécifiques ont pu être déterminées. Suite à cette tâche, cinq nouvelles variables ont été ajoutées à la population : la présence ou l'absence de chacune des cinq pathologies considérées. Ceci clôture la génération de la population synthétique pour l'année initiale.

5.1.2 Évolution dynamique de la population générée

Cette étape est essentielle au projet puisqu'elle est nécessaire pour réaliser les différents exercices prospectifs attendus. Elle concerne l'évolution temporelle de la population synthétique grâce à la modélisation de plusieurs dynamiques. Cette section vise donc à expliquer brièvement les processus qui, en tenant compte des statistiques existantes, donnent lieu aux naissances, aux décès, aux mariages, aux changements de statut socio-professionnel et autres évolutions. En effet, l'évolution de la population synthétique doit être la plus proche possible de la réalité. Ces implémentations sont compliquées parce qu'elles demandent des compromis entre simplicité et proximité à la réalité. Des dynamiques trop complexes ne peuvent être implémentées car elles demandent trop de ressources, principalement au niveau du temps de réalisation. Pour contourner cette difficulté, les chercheurs procèdent par paliers en développant, dans un premier temps, des algorithmes assez simples. Ceux-ci sont ensuite progressivement améliorés pour s'approcher au mieux de la réalité en fonction du temps restant.

Nous allons maintenant expliquer brièvement les différentes dynamiques qui permettent l'évolution temporelle de la population virtuelle. Pour nous aider, le graphique 5.1 indique les processus implémentés dans VBIH ainsi que leur ordre d'apparition au cours d'une année. Commençons par des dynamiques plutôt simples. Après une année, nous avons tous eu notre anniversaire, nous vieillissons d'un an. Au cours de celle-ci, des personnes vont décéder mais d'autres vont naître. Il faudra donc penser aux méthodes liées aux naissances et aux décès. De plus, les individus vont évoluer dans la vie professionnelle. Ils peuvent quitter le domicile parental et entrer dans la vie active. Au cours d'une année, les plus jeunes, généralement, peuvent obtenir un nouveau diplôme tandis que les plus âgés ont la possibilité, après un certain âge, de partir à la retraite. Un changement de statut professionnel, ou un déménagement, doit être envisagé pour toutes personnes se situant dans la vie active. Enfin, certaines dynamiques sont de l'ordre du relationnel : pendant l'année, des couples peuvent se marier et d'autres divorcer. Certaines évolutions étant plus complexes que d'autres, les chercheurs ont fait face à quelques difficultés, notamment avec le module du mariage.

Un point d'attention à prendre en compte est l'ordre dans lequel les dynamiques apparaissent au cours d'une année. Selon cet ordre, les résultats peuvent être différents et se rapprocher ou non de la réalité. Certaines conditions doivent donc être respectées. Un individu ne peut pas se marier et divorcer la même année par exemple. Une autre restriction est l'ordre d'apparition des évènements naissances et décès car une personne décédée peut avoir accouché plus tôt dans l'année et un bébé peut mourir l'année de sa naissance. Cela signifie que la dynamique des naissances doit survenir avant celle des décès mais après le vieillissement de la population. Dans le cas contraire, lors de la première évolution des individus, la population contiendrait trop de bébés âgés de 0 an et cela aurait des impacts sur les analyses prospectives.

L'article [NAX2016] détaille les dynamiques présentes sur l'image 5.1 en donnant pour chaque processus, les méthodes utilisées et les données utiles à leur réalisation. Pour toutes les dynamiques, des explications sont écrites sur ce qui est en cours de réalisation ainsi que sur les modifications qui peuvent

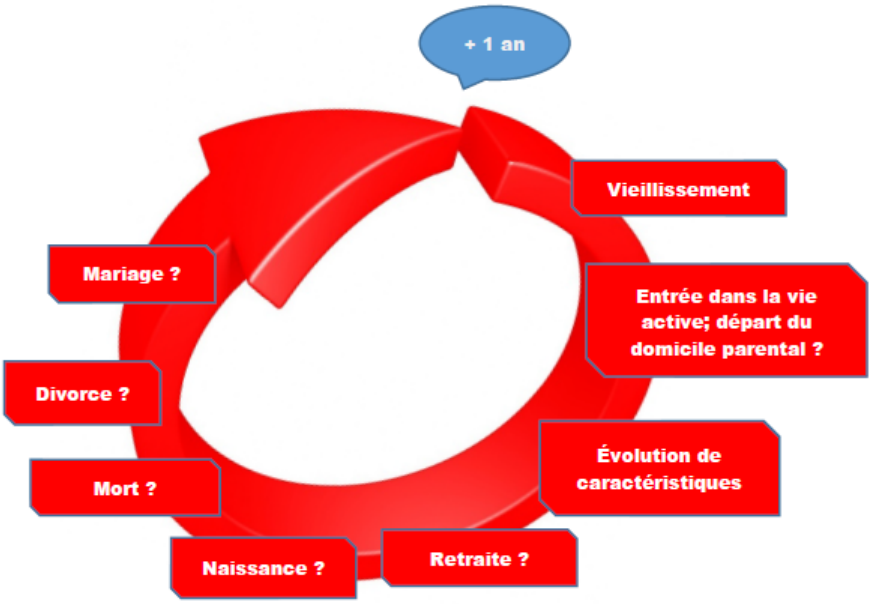


FIGURE 5.1 – Schéma des processus d'évolution temporelle de la population synthétique, provenant de la ressource [NAX2016]

être effectuées plus tard. Concentrons-nous sur le processus des mariages. Nous avons rassemblé ces informations dans le tableau 5.1, qui se base sur l'article [NAX2016].

Grâce à Mme Morgane Dumont, chercheuse faisant partie du projet VBIH, nous pouvons détailler l'implémentation du module concernant les mariages à ce jour. Tout d'abord, cette dynamique est effectuée en plusieurs étapes. Afin que deux individus se marient et forment un nouveau ménage, ils doivent passer par plusieurs stades ; le premier concerne la décision de se marier cette année ou de rester célibataire, le deuxième, le couplage des individus deux par deux. La dernière étape est la modification ou création de caractéristiques pour les ménages engendrés. Plusieurs modèles dynamiques sont proposés

Processus dynamiques	**Méthode**	**Données nécessaires**
Choix de se marier	**Phase 1** : Considérer les taux de mariage par âge et genre. **Phase 2** : Tester un choix discret.	**Phase 1** : Nombre de mariages par âge et genre pour plusieurs années (prendre en compte le nombre de mariages pour une année à la fois). **Phase 2** : Données précises sur les genres, diplômes, statut d'activités, ... par couple.
Avec qui se marier	**Hypothèses** : Seuls les célibataires vont se marier. On ne considère pas les cas de divorce et remariage la même année. **Phase 1** : Considérer les couples à travers toute la Belgique (pas les communes), en ne regardant que les tables d'âges des deux partenaires + changement type et taille du ménage. **Phase 2** : Ajouter le niveau local et faire un choix discret si possible. Ajout des mariages de même genre ? Simplement un 'bruit' ? On pourrait tirer aléatoirement un nombre de 'candidats' pour le mariage, leur attribuer des scores (sur base de la commune, du diplôme etc.) pour ensuite garder le meilleur.	**Phase 2** : Ces tables par commune et pour chaque mariage, les caractéristiques de l'homme et de la femme lors des dernières années.
Suite au mariage, emménagement ? Où ?	Voir ce qui serait le plus cohérent : dans le voisinage de l'un ou de l'autre, dans une zone urbaine (potentiellement le plus possible de leur lieu de travail), entre les deux communes de départ pour rester équidistants des deux familles.	Commune choisie pour l'emménagement en fonction des deux communes précédentes pour tous les mariages.

TABLE 5.1 – Description des processus liés à la dynamique des mariages

dans la littérature traitant de cette thématique en l'abordant sous différents angles. Nous en avons effectivement relevé une partie à la section 1.2 et en avons proposé certains au cours de notre mémoire. Il faut savoir que, par manque de temps et surtout dû à la complexité de cette dynamique, le processus du mariage au sein de VBIH est toujours à la phase 1.

Détaillons alors cette première phase en commençant avec l'extraction des individus pouvant se marier. Ils ne doivent pas faire partie de ménages collectifs ni autres ; ils doivent être célibataires ou veufs ; ils doivent être âgés de 18 ans ou plus. Pour décider de l'entrée ou non des individus dans le marché du mariage, ils ont utilisé des taux statistiques belges par âge, genre et code INS. Ils voudraient tenir compte des causalités provoquant ce choix et alors améliorer le processus grâce aux modèles de choix discrets. D'autres dynamiques sont concernées par cette théorie des choix discrets comme la décision de divorcer. Ce dernier modèle est de haute qualité car les données nécessaires à son calibrage étaient disponibles.

La deuxième étape, celle où les partenaires sont assemblés, s'effectue de la sorte. Pour déterminer le nombre de mariages possible, ils utilisent la valeur minimale entre le nombre d'hommes et de femmes dans le marché du mariage. Ensuite, seule la différence d'âge est prise en compte pour former le couplage. Ils ont utilisé des taux statistiques belges sur la différence entre l'âge des partenaires mariés et un générateur de nombres aléatoires pour décider, pour chaque homme, la meilleure épouse. Si plusieurs femmes possèdent l'âge obtenu, la première rencontrée est choisie. Il est possible qu'aucune épouse ne soit trouvée avec cette caractéristique. L'homme reste alors célibataire une année supplémentaire. Une remarque importante concerne leur parallélisation du code. Les chercheurs du projet VBIH ont décidé de paralléliser leur code en fonction des communes belges. Une conséquence de cette décision est l'incapacité d'interaction entre les individus de deux codes INS différents. Cela signifie, dans le cadre de notre dynamique, que deux individus provenant de deux communes différentes ne peuvent pas se rencontrer.

La dernière étape est la mise à jour des caractéristiques des nouveaux ménages formés. La fusion des deux ménages des partenaires se réalise dans celui de l'individu 1, soit celui de l'homme. Ce n'est donc pas une création mais bien une mise à jour d'un foyer. La femme part alors vivre dans la maison de son mari. Leur statut marital doit être changé en marié. Ensuite, la même opération est réalisée sur les deux partenaires et concerne les autres individus de leur ancien foyer. Si la personne se mariant est le chef de ménage d'un foyer composé de deux personnes ou plus ; si, de plus, son ménage n'est pas un mono-parental, pour chaque membre de celui-ci, les chercheurs vérifient d'abord le lien avec le chef de ménage. Si ce n'est ni le chef de ménage, ni son enfant et si il est âgé d'au moins 18 ans, il devient chef de ménage à la place de l'individu se mariant et forme un nouveau ménage. Une fois un chef de ménage trouvé, toute autre personne n'emménageant pas avec le couple marié ne peut plus prendre cette place et rejoint le nouveau foyer. Une procédure différente est mise en place si c'est un enfant qui quitte ses parents pour se marier. Toutes les personnes au sein de son ancien foyer sont injectées dans un nouveau ménage.

Concluons, à présent, sur le module du mariage de VBIH implémenté actuellement. Bien que les chercheurs se soient arrêtés en phase 1, des idées intéressantes sont mises en place. En effet, à cause de la complexité de ce module, des modèles assez simples doivent être travaillés dans un premier temps. Malheureusement, leur premier jet n'est pas assez convaincant puisqu'il produit un seul mariage.

5.2 Intégration de nos résultats

Après avoir implémenté plusieurs solutions pour reproduire les mariages belges dans une population synthétique, il est intéressant d'étudier leur comportement dans une situation pratique. Malheureusement, ce travail est de grande envergure puisqu'il faut comprendre, dans un premier temps, le code d'une personne extérieure et dans un second temps, adapter ce code pour y intégrer nos modules. Nous n'avons, en conséquence, pas eu l'opportunité de réaliser cet effort. Toutefois, nous trouvons intéressant de simplifier le travail pour une tierce personne qui voudrait s'y atteler. Nous avons alors décrit la manière dont nous intégrerions les algorithmes au sein du code C++ de VirtualBelgium In Health.

Nous commençons déjà par un avantage avec cette population synthétique : leur module des mariages est divisé en trois étapes distinctes comme nous l'avons imaginé dans ce mémoire. La première tâche est la création du marché du mariage. Suite à l'analyse à la section 3.3, nous tenterions d'insérer un réseau de neurones dans ce code en C++. Plusieurs choix s'offrent alors à nous, il faudrait les étudier pour conclure sur leur robustesse mais aussi sur leur temps de résolution. Le premier serait de créer un réseau de neurones dans le langage C++ en définissant les mêmes méta-paramètres que nous avons utilisés en Python. Nous ne nous attendons pas à avoir les mêmes probabilités mais des semblables. Une autre possibilité est d'intégrer notre code Python au sein du code C++. Cette proposition nous laisse toutefois sceptique puisque nous pensons que le temps de résolution du réseau de neurones serait assez long. Il serait quand même intéressant de l'analyser car nous devrions retrouver les probabilités obtenues précédemment. Nous ne ferions donc pas face à de mauvais résultats. La dernière solution est d'effectuer le codage du réseau manuellement en C++ grâce aux valeurs des poids et autres paramètres obtenus. Cette idée est assez fastidieuse mais permet d'acquérir des résultats identiques à ceux étudiés précédemment. Nous devons enfin être vigilants aux données en présence. En effet, bien que les données de VBIH possèdent les informations concernant le groupe d'âges des membres des ménages, les informations sur leur diplôme diffèrent des nôtres. En effet, ils utilisent la classification internationale type de l'éducation (CITE) alors que nous ne possédons que quatre catégories dans nos données. De plus, peu d'individus possèdent un diplôme indéterminé ; par exemple, à Anvers, cela concerne 3% des habitants. Dans nos données, nous avons une majorité d'indétermination. Nous ne pouvons donc pas appliquer notre modèle sur ces données sans traitement au préalable.

Ensuite, pour coupler les individus deux à deux au sein du marché des mariages, nous pouvons insérer la méthode probabiliste implémentée au chapitre 4, apportant un couplage total des personnes éligibles. De nouveau, nous avons utilisé un réseau de neurones pour déterminer les préférences des hommes envers les femmes du marché. Une condition doit être ajoutée si le nombre d'individus de chaque genre n'est pas équilibré. Nous pouvons, entre autres, décider d'écarter, de manière aléatoire, autant de personnes que nécessaire du genre dominant en effectif.

Finalement, nous avons peu abordé la dernière étape pour obtenir une solution complète : la mise à jour des caractéristiques des nouveaux ménages. Dans un premier temps, il serait préférable de ne pas modifier le code de VBIH pour cette phase. En effet, plusieurs paramètres sont déjà pris en compte. Si le temps est disponible, il faudrait améliorer le changement des liens avec le chef de ménage. En effet, seules les conjointes voient leur lien modifié. Il n'en est rien pour les autres individus des deux anciens foyers. Ensuite, il peut être intéressant de modifier le processus de l'emménagement car actuellement, toutes les femmes se mariant vont obligatoirement vivre dans le ménage masculin et ne sont jamais chefs de ménage.

Conclusion

Nous désirons conclure notre mémoire en résumant le travail effectué mais aussi en décrivant quelques perspectives et ouvertures.

Nous avons débuté notre réflexion, au premier chapitre, en étudiant plusieurs modèles de simulation des mariages provenant de la littérature. Ce travail a permis de dégager quelques idées pour la suite, notamment l'utilisation de modèles de choix discrets ou encore l'affaiblissement des critères de recherche du partenaire idéal. Après une étude statistique sommaire, présente au chapitre 2, nous avons entamé le sujet principal de ce mémoire, soit l'implémentation d'une solution modélisant la dynamique des mariages au sein de la population belge. Nous avons décidé de réaliser ce travail en deux parties distinctes. Ce choix est motivé par l'étude de la littérature et l'indépendance des deux modules. Le premier forme le marché des mariages, un ensemble fictif groupant tous les individus désirant se marier au cours d'une année fixée. La construction de ce marché est accomplie au cours du chapitre 3. Le second module, au chapitre 4, forme les couples parmi les individus présents dans le marché. Nous allons, à présent, résumer les informations concernant ces deux parties de solution.

Pour construire le marché des mariages, nous avons utilisé deux concepts théoriques : les modèles de choix discrets et les réseaux de neurones. Les modèles de choix ne retournaient pas de bons résultats - les individus au sein du marché ne possédaient pas les mêmes caractéristiques démographiques que la population dont le mariage a été officialisé en 2000. Nous pensons que la cause principale de ce problème est le manque d'informations sur les individus dans les données. Par contre, nous sommes très satisfaits du réseau de neurones implémenté. Le marché formé contient seulement quelques individus de plus par rapport à la réalité. Cette constatation n'est pas négative. En effet, il est préférable d'obtenir un marché des mariages plus fourni puisqu'en effectuant le couplage, il est possible que certaines personnes ne trouvent pas de partenaire.

Le deuxième module, l'implémentation du couplage, se déroule en deux étapes. Nous avons opté pour une solution homme-dominant où ces derniers choisissent leur femme parmi celles présentes dans le marché des mariages. Ne possédant aucune information sur les caractéristiques idéales des épouses pour chaque homme, nous avons construit ces préférences grâce à deux méthodes théoriques, les choix discrets et les réseaux de neurones. De nouveau, de meilleurs résultats étaient observés chez ces derniers. De cet algorithme est retourné, pour chaque homme, la probabilité de prendre pour épouse chaque femme du marché. Cette première partie de solution est insérée dans la seconde où le matching est produit. Un premier modèle assez simpliste est proposé. Nous l'avons amélioré de deux manières différentes pour obtenir, au final, une méthode offrant un couplage total des individus. Cette dernière proposition offre de bonnes prédictions, des mariages aux caractéristiques semblables à la réalité. Nous avons aussi étudié une solution reposant sur la théorie du recuit simulé. Malheureusement, cette méthode, en plus d'être très lente, n'amène pas de bons résultats au niveau de la distribution des différences d'âges des partenaires au sein des couples. Elle produit quelques mariages extrêmes.

Le dernier chapitre de notre mémoire est une aide pour une perspective de travail en collaboration

avec le projet VirtualBelgium In Health. Nous ne voulons pas que nos efforts soient vains. Nous voulons observer les résultats suite à l'implantation de notre solution en situation réelle. Malheureusement, cette tâche est trop importante dans le cadre de notre rédaction et pourrait justement être sujet d'un futur mémoire. Plusieurs modifications importantes doivent être accomplies avant de pouvoir intégrer nos algorithmes au sein des codes C++ de la population synthétique de VBIH. Il faut notamment penser aux divergences d'informations entre nos données et celles du projet, pour les catégories de diplômes par exemple. Cette perspective reste néanmoins intéressante et stimulante pour nous.

D'autres ouvertures peuvent être proposées. Nos deux réseaux de neurones peuvent être étudiés en profondeur en jouant sur les méta-paramètres explorés ou d'autres. Si des ordinateurs puissants sont disponibles, nous recommandons l'utilisation de la fonction GridSearchCV avec plusieurs méta-paramètres et plusieurs valeurs de ceux-ci. De cette manière, le modèle optimal sera obtenu. Une autre perspective est l'étude d'autres modèles théoriques pour chaque module de notre solution. Nous pourrions envisager les méthodes des mariages stables ou encore les algorithmes génétiques, qui appartiennent aux méthodes probabilistes. Enfin, il serait intéressant d'étudier ces modules sur d'autres données, provenant de populations d'autres pays. Pour envisager cette exploration, il est important de savoir que les jeux de données concernés ne sont pas souvent disponibles en libre accès.

Bibliographie

[BEN2009] Benaych-Georges, F., *Une brève explication du principe de fonctionnement de l'algorithme de recuit simulé*, LPMA, UPMC Univ Paris, CMAP, École Polytechnique, Paris, France, 2009.

[BIE2007] Bierlaire, M., *BIOGEME : an open source package for the estimation of advanced discrete choice models transp-or.epfl.ch*, Transport and Mobility Laboratory, Ecole Polytechnique Fedérale de Lausanne, Lausanne, Swiss, 2007.

[BIE2008] Bierlaire, M., *Estimation of discrete choice models with BIOGEME 1.6*, Transport and Mobility Laboratory, Ecole Polytechnique Fedérale de Lausanne, Lausanne, Swiss, 2008.

[BIE2009] Bierlaire, M., Fetiarison, M., *Estimation of discrete choice models : extending BIOGEME*, Transport and Mobility Laboratory, Ecole Polytechnique Fedérale de Lausanne, Lausanne, Swiss, 2009.

[BIE2016] Bierlaire, M., *PythonBiogeme : a short introduction.*, Report TRANSP-OR 160706, Series on Biogeme, Transport and Mobility Laboratory, Ecole Polytechnique Fedérale de Lausanne, Lausanne, Swiss, 2016.

[BIL2007] Billari, F., Prskawetz, A., Diaz, B., Fent, T., *The "Wedding-Ring" : An Agent-Based marriage model based on social interaction*, Demographic Research, 17(3), 59-82, 2007.

[BLA2016] Blanchard, J., Dumont, M., Petit, A., *TPP Master en sciences mathématiques : Une population synthétique pour la province de Namur PARTIE 1*, Unamur, Namur, Belgique, 2016.

[CAL1997] Caldwell, S., *Chapter 5 - CORSIM*, Society of Actuaries, Cornell University, New york, États-Unis, 1997.

[CHA2001] Chaib-draa, B., Jarras, I., Moulin, B., *Systèmes multiagents : Principes généraux et applications* , Agent et systèmes multiagents, Hermès, 2001.

[CHO2012] Chorus, C., *A Random Regret Minimization-based Discrete Choice Model*, Random Regret-based discrete choice modeling : A tutorial, Springer, Heidelberg, Germany, 2, 11-24, 2012.

[CHO2013] Chorus, C., Rose, J., *Selecting a date : A matter of regret and compromises*, Choice Modelling, 11, 229-242, 2013.

[CHO2014(1)] Chorus, C., *Capturing alternative decision rules in travel choice models : a critical discussion*, Handbook of choice modelling, 13, 290-310, 2014.

[CHO2014(2)] Chorus, C., Van Cranenburgh, S., Dekker, T., *Random regret minimization for consumer choice modeling : Assessment of empirical evidence*, Journal of Business Research, 67, 2428-2436, 2014.

[COR2015] Cornelis, E., Barthelemy, J., *A synthetic population for Luxembourg and its demographic evolution*, Report naXys, 2015.

[COR2017] Cornelis, E., Dumont, M., Bataille, M., *Choix discrets et autres approches mathématiques pour la mobilité*, UNamur, Namur, 2017.

[DEB2005] Debrand, T., Taffin, C., *Les facteurs structurels et conjoncturels de la mobilité résidentielle depuis 20 ans*, Économie et Statistique, 381-382, 125-146, 2005.

[DEB2006] Debrand, T., Taffin, C., *Les changements de résidence : entre contraintes familiales et professionnelles*, Données sociales : La société française, 505-513, 2006.

[DEB2009] Deboosere, P., et al., *Ménages et familles en Belgique*,Enquête Socio-économique 2001 Monographies, Direction générale Statistique et Information économique, Bruxelles, 4, 2009.

[DUM2017] Dumont, M., Barthélemy, J., Carletti, T., *Robustness of artificial neural network and discrete choice modelling in presence of unobserved variables*, 22nd International Congress on Modelling and Simulation (MODSIM2017), UNamur, Namur, 2017.

[ELL2017] Elloumi, S., *Le recuit simulé*, IMO, RCP104 : Optimisation en informatique, Conservatoire national des arts et métiers, Paris, France, 2017.

[FEI1998] Fei, J., *Chapitre 6 : Les modèles de choix discrets et leur estimation*, Choix modal et système logistique en transport de marchandises : Modélisation, Analyse Economique et Prévision du Comportement du Chargeur, École Nationale des Ponts et Chaussées, 84-102, 1998.

[FRE2017] Frenay, B., *Machine learning et data mining*, UNamur, Namur, 2017.

[GEA2013] Geard, N., *Synthetic Population Dynamics : A Model of Household Demography*, Journal of Artificial Societies and Social Simulation, 16 (1) 8, 2013.

[HIL2008] Hills, T., Todd, P., *Population Heterogeneity and Individual Differences in an Assortative Agent-Based Marriage and Divorce Model (MADAM) Using Search with Relaxing Expectations*, Journal of Artificial Societies and Social Simulation, 11(4), 5, 2008.

[HUE2013] Huet, S., *Individual based models of social systems : data driven hybrid micro-models of rural development and collective dynamics of filtering or rejecting messages*, Université Blaise Pascal - Clermont II, Clermont-Ferrand, France, 2013.

[HUE2012] Huet, S., et al., *Micro-simulation model of municipality network in the Auvergne case study*, PRIMA working paper, Aubière, France, 2012.

[KIN1999] King, A., Bækgaard, H., Robinson, M., *DYNAMOD-2 : An overview*, Technical Paper n° 19, NATSEM, Université de Canberra, Australie, 1999.

[MOR2010] Morand, E., et al., *Demographic modelling : the state of the art*, SustainCity Working Paper, 2.1a, Ined, Paris, 2010.

[MOR2000] Morrison, R., *DYNACAN, the Canada Pension Plan Policy Model : Demographics and Earnings Components*, in Gupta, A., Kapur, V. (eds), Microsimulation in Government Policy and Forecasting, North-Holland, Amsterdam, 341-360, 2000.

[MUD2015] Mudimu, E., Engelbrecht, G., *Agent-based model for social and sexual partnerships formation*, Adaptive Behavior, 23(1), 34-49, 2015.

[NAM2014] Namazi-Rad, M., Mokhtarian, P., Perez, P., *Generating a Dynamic Synthetic Population – Using an Age-Structured Two-Sex Model for Household Dynamics*, PLOS ONE, 9(7), 1-16, 2014.

[NAX2016] Naxys, DEMO, *VirtualBelgium in Health : Une plateforme de population virtuelle comme outil d'aide à la décision en matière de prospective et de planification des besoins de soins de santé pour les personnes âgées*, rapport de mi-parcours non-publié, UNamur, Namur, Belgique, et UCL, Louvain-la-neuve, Belgique, 2016.

[NOL2013] Nollet, J., et al., *VirtualBelgium in Health : Une plateforme de population virtuelle comme outil d'aide à la décision en matière de prospective et de planification des besoins de soins de santé pour les personnes âgées*, convention n°1318077, Namur, Belgique, 2013.

[ORC1976] Orcutt, G., et al., *Policy exploration through microanalytic simulation*, Urban Institute Press, Washington, États-Unis, 1976.

[PIC2014] Picard, G., *VirtualBelgium : Modélisation et Simulation des mariages et divorces*, Université de Namur, Namur, Belgique, 2014.

[PEI2017] Pei, X., Zhan, X., Jin, Z., *Application of pair approximation method to modeling and analysis of a mariage network*, Applied Mathematics and Computation, 294, 280-293, 2017.

[REB2015] Reboul, L., *C- Liaison entre deux variables statistiques*, Département de mathématiques, Faculté de Luminy, Marseille, France, 2015.

[SPI2013] Spielauer, M., et al., *Modèle de micro-simulation LifePaths : Vue d'ensemble*, Statistique Canada – Division de la modélisation, Ottawa, Ontario, 2013.

[VIL2013] Villers, P., Gillin, A., *Programme WB Health (proposition détaillée) - Projet VBIH*, Service public de Wallonie, Jambes, Belgique, 2013.

[ZAD2001] Zaidi, A., Rake, K., *Dynamic Microsimulation Models : A Review and Some Lessons for SAGE*, ESRC-Sage discussion papers, Vol. 2, Londres, Royaume-Uni, 2001.

[ZIN2012] Zinn, S., *A Mate-Matching Algorithm for Continuous-Time Microsimulation Models*, International Journal Of Micro-simulation, 5(1), 31-51, 2012.

Annexe A

Annexes

A.1 Fichiers et variables utilisés concernant la population belge

Fichier	Variable	Informations
couples2001.txt		mariages belges enregistrés au 1er Octobre 2001
	sexecm	sexe du chef de ménage 1=homme ; 2=femme
	sexecj	sexe du conjoint 1=homme ; 2=femme
	agecm	âge du chef de ménage
	agecj	âge du conjoint
suivicouples.txt		mariages belges enregistrés au 1er Octobre 2001
	sexecm	sexe du chef de ménage 1=homme ; 2=femme
	genercm	catégorie d'âges du chef de ménage par tranche de 5 ans
	genercj	catégorie d'âges du conjoint par tranche de 5 ans
	acheciv	année de mariage du couple
	dipglcm	diplôme du chef de ménage indéterminé, primaire, secondaire ou supérieur
	dipglcj	diplôme du conjoint indéterminé, primaire, secondaire ou supérieur
recens2001.txt		caractéristiques de la population belge au 1er Octobre 2001
	grage	catégorie d'âges de l'individu par tranche de 5 ans
	sexe	sexe de l'individu 1=homme ; 2=femme
	dipgl	diplôme de l'individu indéterminé, primaire, secondaire ou supérieur
	COUNT	nombre d'individus possédant ces caractéristiques
recens2001etendu.txt		fichier *recens2001.txt* où chaque ligne correspond à un seul individu

TABLE A.1 – Informations techniques sur les fichiers et les variables utilisés

A.2 Table de valeurs de la loi khi-carré

	α										
ddl	0.995	0.990	0.975	0.950	0.900	0.500	0.100	0.050	0.025	0.010	0.005
1	0.00	0.00	0.00	0.00	0.02	0.45	2.71	3.84	5.02	6.63	7.88
2	0.01	0.02	0.05	0.10	0.21	1.39	4.61	5.99	7.38	9.21	10.60
3	0.07	0.11	0.22	0.35	0.58	2.37	6.25	7.81	9.35	11.34	12.84
4	0.21	0.30	0.48	0.71	1.06	3.36	7.78	9.94	11.14	13.28	14.86
5	0.41	0.55	0.83	1.15	1.61	4.35	9.24	11.07	12.83	15.09	16.75
6	0.68	0.87	1.24	1.64	2.20	5.35	10.65	12.59	14.45	16.81	18.55
7	0.99	1.24	1.69	2.17	2.83	6.35	12.02	14.07	16.01	18.48	20.28
8	1.34	1.65	2.18	2.73	3.49	7.34	13.36	15.51	17.53	20.09	21.96
9	1.73	2.09	2.70	3.33	4.17	8.34	14.68	16.92	19.02	21.67	23.59
10	2.16	2.56	3.25	3.94	4.87	9.34	15.99	18.31	20.48	23.21	25.19
11	2.60	3.05	3.82	4.57	5.58	10.34	17.28	19.68	21.92	24.72	26.76
12	3.07	3.57	4.40	5.23	6.30	11.34	18.55	21.03	23.34	26.22	28.30
13	3.57	4.11	5.01	5.89	7.04	12.34	19.81	22.36	24.74	27.69	29.82
14	4.07	4.66	5.63	6.57	7.79	13.34	21.06	23.68	26.12	29.14	31.32
15	4.60	5.23	6.27	7.26	8.55	14.34	22.31	25.00	27.49	30.58	32.80
16	5.14	5.81	6.91	7.96	9.31	15.34	23.54	26.30	28.85	32.00	34.27
17	5.70	6.41	7.56	8.67	10.09	16.34	24.77	27.59	30.19	33.41	35.72
18	6.26	7.01	8.23	9.39	10.87	17.34	25.99	28.87	31.53	34.81	37.16
19	6.84	7.63	8.81	10.12	11.65	18.34	27.20	30.14	32.85	36.19	38.58
20	7.43	8.26	9.59	10.85	12.44	19.34	28.41	31.41	34.17	37.57	40.00
21	8.03	8.90	10.28	11.59	13.24	20.34	29.62	32.67	35.48	38.93	41.40
22	8.64	9.54	10.98	12.34	14.04	21.34	30.81	33.92	36.78	40.29	42.80
23	9.26	10.20	11.69	13.09	14.85	22.34	32.01	35.17	38.08	41.64	44.18
24	9.89	10.86	12.40	13.85	15.66	23.34	33.20	36.42	39.36	42.98	45.56
25	10.52	11.52	13.12	14.61	16.47	24.34	34.28	37.65	40.65	44.31	46.93
26	11.16	12.20	13.84	15.38	17.29	25.34	35.56	38.89	41.92	45.64	48.29
27	11.81	12.88	14.57	16.15	18.11	26.34	36.74	40.11	43.19	46.96	49.65
28	12.46	13.57	15.31	16.93	18.94	27.34	37.92	41.34	44.46	48.28	50.99
29	13.12	14.26	16.05	17.71	19.77	28.34	39.09	42.56	45.72	49.59	52.34
30	13.79	14.95	16.79	18.49	20.60	29.34	40.26	43.77	46.98	50.89	53.67
40	20.71	22.16	24.43	26.51	29.05	39.34	51.81	55.76	59.34	63.69	66.77
50	27.99	29.71	32.36	34.76	37.69	49.33	63.17	67.50	71.42	76.15	79.49
60	35.53	37.48	40.48	43.19	46.46	59.33	74.40	79.08	83.30	88.38	91.95
70	43.28	45.44	48.76	51.74	55.33	69.33	85.53	90.53	95.02	100.42	104.22
80	51.17	53.54	57.15	60.39	64.28	79.33	96.58	101.88	106.63	112.33	116.32
90	59.20	61.75	65.65	69.13	73.29	89.33	107.57	113.14	118.14	124.12	128.30
100	67.33	70.06	74.22	77.93	82.36	99.33	118.50	124.34	129.56	135.81	140.17

A.3 Préparation des données pour la première étude statistique

Preparation des donnees pour l'analyse de l'influence des variables sur la decision d'un individu de se marier au cours d'une annee

May 24, 2018

```
In [ ]: #Packages utiles
        import pandas as pd
        import numpy as np
```

1 Personnes s'étant mariées en 2000

1.1 Chargement des données

```
In [ ]: df_married = pd.read_csv("Donnees//suivicouples.txt", sep="\t")
```

1.2 Transformation des données

```
In [ ]: df_CD1 = df_married[df_married.acheciv==2000]
```

```
In [ ]: #Remplacement des nan afin d'éviter les erreurs
        df_CD1 = df_CD1.replace(np.nan, 'IND', regex=True)
```

```
In [ ]: #Index corrects
        df = df_CD1.reset_index()
```

1.3 Transformation pour obtenir une ligne par individu

```
In [ ]: #Construction de deux df : un pour le chef de ménage et un pour le conjoint
        df_cm = df[['sexecm', 'genercm', 'dipglcm']]
        df_cm.columns = ['sexe', 'grage', 'diploma']

        df_cj = df[['sexecm', 'genercj', 'dipglcj']]
        df_cj.columns = ['sexe', 'grage', 'diploma']
```

```
In [ ]: #Sexe du conjoint est l'opposé de celui du chef de ménage
        df_cj['sexe'] = df_cj['sexe'].replace([1, 2], [2, 1])
```

```
In [ ]: #Fusion des deux df
        df = pd.concat([df_cm, df_cj])
```

1.4 Transformation des classes en entiers

```
In [ ]: classe_diploma = ['IND', 'PRI', 'SEC', 'SUP']

        classe_diploma_trad = []
        for i in range(len(classe_diploma)):
            classe_diploma_trad.append(i)

        df['diploma'] = df['diploma'].replace(classe_diploma, classe_diploma_trad)
```

```
In [ ]: classe_age = ['00-05', '05-10', '10-15', '15-20', '20-25', '25-30', '30-35',
                       '35-40', '40-45', '45-50', '50-55', '55-60', '60-65', '65-70',
                       '70-75', '75-80', '80-85', '85-90', '90-95', '95-100', 'plus de 100']

        classe_age_trad = []
        for i in range(len(classe_age)):
            classe_age_trad.append(i)

        df['grage'] = df['grage'].replace(classe_age, classe_age_trad)
```

```
In [ ]: df.to_csv('Donnees//data_CD1.csv', sep=' ', header=True, index=False)
```

2 Célibataires en 2001

2.1 Chargement des données

```
In [ ]: df_tot = pd.read_csv("Donnees//recens2001etendu.txt", sep="\t")
```

2.2 Transformation des données

```
In [ ]: #Retrait des enfants
        df_adult = df_tot[(df_tot.grage2001!='00-05') & (df_tot.grage2001!='05-10')
                          & (df_tot.grage2001!='10-15')]
        df_adult = df_adult[['grage2001', 'sexe', 'dipgl']]
        df_adult.columns = ['grage', 'sexe', 'diploma']
```

```
In [ ]: classe_diploma = ['IND', 'PRI', 'SEC', 'SUP']

        classe_diploma_trad = []
        for i in range(len(classe_diploma)):
            classe_diploma_trad.append(i)

        df_adult['diploma'] = df_adult['diploma'].replace(classe_diploma,
                                                          classe_diploma_trad)
```

```
In [ ]: classe_age = ['00-05', '05-10', '10-15', '15-20', '20-25', '25-30', '30-35',
                       '35-40', '40-45', '45-50', '50-55', '55-60', '60-65', '65-70',
                       '70-75', '75-80', '80-85', '85-90', '90-95', '95-100', 'plus de 100']
```

```
        classe_age_trad = []
        for i in range(len(classe_age)):
            classe_age_trad.append(i)

        df_adult['grage'] = df_adult['grage'].replace(classe_age, classe_age_trad)
```

2.3 Retrait des personnes mariées pour conserver uniquement les célibataires

```
In [ ]: df_married = df_married.replace(np.nan, 'IND', regex=True)

        df_cm = df_married[['genercm', 'sexecm', 'dipglcm']]
        df_cm.columns = ['grage', 'sexe', 'diploma']

        df_cj = df_married[['genercj', 'sexecm', 'dipglcj']]
        df_cj.columns = ['grage', 'sexe', 'diploma']
        df_cj['sexe'] = df_cj['sexe'].replace([1, 2], [2, 1])

        df_married = pd.concat([df_cm, df_cj])

In [ ]: classe_diploma = ['IND', 'PRI', 'SEC', 'SUP']

        classe_diploma_trad = []
        for i in range(len(classe_diploma)):
            classe_diploma_trad.append(i)

        df_married['diploma'] = df_married['diploma'].replace(classe_diploma,
                                                              classe_diploma_trad)

In [ ]: classe_age = ['00-05', '05-10', '10-15', '15-20', '20-25', '25-30', '30-35',
                       '35-40', '40-45', '45-50', '50-55', '55-60', '60-65', '65-70',
                       '70-75', '75-80', '80-85', '85-90', '90-95', '95-100', 'plus de 100']

        classe_age_trad = []
        for i in range(len(classe_age)):
            classe_age_trad.append(i)

        df_married['grage'] = df_married['grage'].replace(classe_age, classe_age_trad)

In [ ]: df_single = df_adult[~df_adult.index.isin(df_married.index)]

In [ ]: df_single.to_csv('Donnees//data_single.csv', sep=' ', header=True, index=False)
```

A.4 Analyse statistique des données relatives à la décision de se marier

Statistiques sur la population decidant de se marier en 2000

May 24, 2018

```
In [1]: #Packages utiles
        import pandas as pd
        import matplotlib.pyplot as plt
        import matplotlib as mpl
        from scipy.stats import chi2_contingency
```

```
In [2]: #Taille du texte pour les graphes
        plt.rc('font', size=18)
```

1 Chargement des données

```
In [3]: #Chargement des données des individus mariés en 2000
        df_2000 = pd.read_csv('Donnees//data_CD1.csv', sep=' ')
```

```
In [4]: #Chargement des données des célibataires
        df_single = pd.read_csv('Donnees//data_single.csv', sep=' ')
```

2 Analyse du sexe

```
In [5]: #Somme de df_2000 par sexe
        df_2000_sexe = df_2000['sexe'].value_counts()

        #Somme de df_single par sexe
        df_single_sexe = df_single['sexe'].value_counts()
```

```
In [9]: #Création d'un tableau croisé avec les valeurs précédentes
        d = {'marié': df_2000_sexe.values, 'célibataire': df_single_sexe.values}
        crosstable = pd.DataFrame(data=d, index=['H', 'F'])
```

```
In [10]: crosstable
```

```
Out[10]:    célibataire  marié
         H      3489336  37332
         F      3270558  37332
```

```
In [11]: #Graphe circulaire
         crosstable.plot(kind='pie', figsize=(12, 5.5), subplots=True, legend=False,
                         colormap='Set2', autopct='%1.0f%%')
         plt.savefig("Stats//pc_sexe.png")
```

```
In [12]: #Test d'indépendance du khi carré
         chi2, pvalue, degrees, expected = chi2_contingency(crosstable)
         print(" Khi carré : ", chi2, "\n", "P-valeur :", pvalue, "\n", "Degrés :",
               degrees, "\n", "Tableau croisé théorique :", expected)
```

```
Khi carré :  77.3659693525
P-valeur : 1.42041603069e-18
Degrés : 1
Tableau croisé théorique : [[ 3488140.98193212    38527.01806788]
[ 3271753.01806788    36136.98193212]]
```

3 Analyse du groupe d'âges

```
In [13]: #Somme de df_2000 par groupe d'âges
         df_2000_age = df_2000['grage'].value_counts()
         df_2000_age[19] = 0
         df_2000_age[20] = 0
         df_2000_age = df_2000_age.sort_index()

         #Somme de df_single par groupe d'âges
         df_single_age = df_single['grage'].value_counts()
         df_single_age = df_single_age.sort_index()
```

```
In [14]: classe_age = ['15-20', '20-25', '25-30', '30-35', '35-40', '40-45', '45-50',
                       '50-55', '55-60', '60-65', '65-70', '70-75', '75-80', '80-85',
                       '85-90', '90-95', '95-100', '100+']

         df_2000_age.index = classe_age
         df_single_age.index = classe_age
```

```
In [15]: #Création d'un tableau croisé avec les valeurs précédentes
         d = {'marié': df_2000_age.values, 'célibataire': df_single_age.values}
         crosstable = pd.DataFrame(data=d, index=df_2000_age.index)
```

```
In [16]: crosstable
```

```
Out[16]:        célibataire  marié
         15-20        481738    789
         20-25        507470  14030
         25-30        519165  26204
         30-35        581380  13332
         35-40        636903   7472
         40-45        628912   4776
         45-50        585782   3345
         50-55        552483   2384
         55-60        457777   1105
         60-65        399948    607
```

```
65-70         400360    311
70-75         376690    159
75-80         310109    101
80-85         174227     34
85-90          96186     12
90-95          40910      3
95-100          8863      0
100+             991      0
```

```
In [20]: #Graphe en bâtonnets
         df_2000_age.plot(kind='bar', figsize=(12, 12))
         plt.xlabel("Groupe d'âges")
         plt.savefig("Stats//bc_age_mar.png")

         df_single_age.plot(kind='bar', figsize=(12, 12))
         plt.xlabel("Groupe d'âges")
         plt.savefig("Stats//bc_age_single.png")
```

```
In [24]: crosstable.plot(kind='pie', figsize=(16, 7.5), labeldistance=1.8,
                         subplots=True, legend=False, colormap='tab20',
                         startangle=90, labels=None, autopct='%1.0f%%',
                         pctdistance=1.1, counterclock=False)

         labels = crosstable.index
         plt.legend(labels, bbox_to_anchor=(1, 0.5), loc="center right",
                    bbox_transform=plt.gcf().transFigure)
         plt.subplots_adjust(right=0.82, top=0.82)
         plt.savefig("Stats//pc_age.png")
```

```
In [23]: chi2, pvalue, degrees, expected = chi2_contingency(crosstable)
         print(" Khi carré : ", chi2, "\n", "P-valeur :", pvalue, "\n", "Degrés :",
               degrees, "\n", "Tableau croisé théorique :", expected)
```

```
Khi carré :  118166.699556
P-valeur : 0.0
Degrés : 17
Tableau croisé théorique : [[  4.77255643e+05   5.27135711e+03]
[  5.15802883e+05   5.69711692e+03]
[  5.39411127e+05   5.95787336e+03]
[  5.88215080e+05   6.49692003e+03]
[  6.37335537e+05   7.03946254e+03]
[  6.26765287e+05   6.92271261e+03]
[  5.82691093e+05   6.43590680e+03]
[  5.48805366e+05   6.06163408e+03]
[  4.53868952e+05   5.01304776e+03]
[  3.96179144e+05   4.37585554e+03]
[  3.96293877e+05   4.37712278e+03]
[  3.72732120e+05   4.11687979e+03]
[  3.06821117e+05   3.38888330e+03]
```

```
[  1.72357289e+05   1.90371101e+03]
[  9.51470868e+04   1.05091324e+03]
[  4.04660467e+04   4.46953297e+02]
[  8.76617632e+03   9.68236764e+01]
[  9.80173839e+02   1.08261608e+01]]
```

4 Analyse du diplôme

```
In [25]: #Somme de df_2000 par diplôme
         df_2000_diploma = df_2000['diploma'].value_counts()
         df_2000_diploma = df_2000_diploma.sort_index()

         #Somme de df_single par diplôme
         df_single_diploma = df_single['diploma'].value_counts()
         df_single_diploma = df_single_diploma.sort_index()

In [26]: classe_diploma = ['IND', 'PRI', 'SEC', 'SUP']

         df_2000_diploma.index = classe_diploma
         df_single_diploma.index = classe_diploma

In [27]: #Création d'un tableau croisé avec les valeurs précédentes
         d = {'marié': df_2000_diploma.values, 'célibataire': df_single_diploma.values}
         crosstable = pd.DataFrame(data=d, index=df_2000_diploma.index)

In [28]: crosstable

Out[28]:      célibataire  marié
         IND       866028  52344
         PRI      1024258   2218
         SEC      3447718  15283
         SUP      1421890   4819

In [31]: crosstable.plot(kind='pie', figsize=(10, 4.5), subplots=True, legend=False,
                         colormap='Set2', startangle=90, labels=None, autopct='%1.0f%%',
                         pctdistance=1.2, labeldistance=1.8, counterclock=False)

         labels = crosstable.index
         plt.legend(labels, bbox_to_anchor=(1, 0.5), loc="center right",
                    bbox_transform=plt.gcf().transFigure)
         plt.subplots_adjust(right=0.83, top=0.83)
         plt.savefig("Stats//pc_dipgl.png")

In [32]: chi2, pvalue, degrees, expected = chi2_contingency(crosstable)
         print(" Khi carré : ", chi2, "\n", "P-valeur :", pvalue, "\n", "Degrés :",
               degrees, "\n", "Tableau croisé théorique :", expected)
```

```
Khi carré :  208815.218799
P-valeur : 0.0
Degrés : 3
Tableau croisé théorique : [[  908339.26240263    10032.73759737]
[ 1015262.28229302    11213.71770698]
[ 3425169.51087312    37831.48912688]
[ 1411122.94443123    15586.05556877]]
```

A.5 Analyse statistique des données relatives aux individus mariés

Statistiques sur les personnes mariees

May 24, 2018

In [1]:
```
#Packages utiles
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib as mpl
import numpy as np
from sklearn import linear_model
from scipy.stats import chi2_contingency
```

In [2]:
```
#Taille du texte pour les graphes
plt.rc('font', size=18)
```

In [3]:
```
#Augmentation du nombre de points sur les graphes
mpl.rcParams['agg.path.chunksize'] = 10000
```

1 Analyse de l'âge

In [4]:
```
#Chargement et transformation des données pour obtenir une ligne par couple
df = pd.read_csv("Donnees//couples2001.txt", sep="\t")
headers = df.columns[0:4]

data_full = []
for i in range(df.shape[0]) :
    for j in range(int(df.COUNT[i])) :
        data_full.append(df.values[i][0:4])

df = pd.DataFrame(data_full, columns=headers)
```

In [5]:
```
#Résumé des données
print(df.describe())
```

```
             sexecm        sexecj         agecm         agecj
count  2.121024e+06  2.121024e+06  2.121024e+06  2.121024e+06
mean   1.015568e+00  1.984432e+00  5.254850e+01  5.024768e+01
std    1.237966e-01  1.237966e-01  1.491728e+01  1.482489e+01
min    1.000000e+00  1.000000e+00  1.900000e+01  1.600000e+01
25%    1.000000e+00  2.000000e+00  4.000000e+01  3.800000e+01
50%    1.000000e+00  2.000000e+00  5.100000e+01  4.900000e+01
```

```
75%    1.000000e+00  2.000000e+00  6.400000e+01  6.200000e+01
max    2.000000e+00  2.000000e+00  1.040000e+02  1.010000e+02
```

```
In [6]: #Distribution de l'âge
        plt.figure(figsize=(12, 8))
        df.boxplot(column=['agecm', 'agecj'], grid=False)
        plt.savefig("Stats//bp_ages.png")
```

```
In [7]: #Nuage de points entre agecm et agecj
        df_ages = df.groupby(df.columns[2:4].tolist()).size().reset_index()
                                .rename(columns={0:'Nbr'})
        s = df_ages.Nbr
        df_ages.plot.scatter('agecm', 'agecj', grid=False, figsize=(12, 8),
                             sharex=False, s=50, c=s, colormap='hot')
        plt.savefig("Stats//sp_ages.png")
```

```
In [8]: #Corrélation
        print(df[df.columns[2:4]].corr())
```

```
          agecm     agecj
agecm  1.000000  0.961288
agecj  0.961288  1.000000
```

```
In [9]: #Régression linéaire
        x = df.agecm.values
        y = df.agecj.values

        x = x.reshape(x.shape[0], 1)
        y = y.reshape(x.shape[0], 1)

        LR = linear_model.LinearRegression()
        LR.fit(x, y)
        print(LR.score(x, y))
        print(LR.coef_)
        print(LR.intercept_)

        plt.figure(figsize=(12, 8))
        plt.xlabel('agecm')
        plt.ylabel('agecj')
        plt.scatter(df_ages.agecm, df_ages.agecj, c=s, cmap='hot', marker='.')
        plt.plot(x, LR.predict(x), color='blue', linewidth=2)
        plt.colorbar()
        plt.savefig("Stats//lr_ages.png")
```

```
0.924074394773
[[ 0.95533441]]
[ 0.04629317]
```

2 Analyse du groupe d'âges

In [10]:
```
#Chargement des données
df = pd.read_csv("Donnees//suivicouples.txt", sep="\t")
```

In [11]:
```
#Transformation du groupe d'âges en entier
classe_age = ['00-05', '05-10', '10-15', '15-20', '20-25', '25-30', '30-35',
              '35-40', '40-45', '45-50', '50-55', '55-60', '60-65', '65-70',
              '70-75', '75-80', '80-85', '85-90', '90-95', '95-100', '100+']

classe_age_trad = []
for i in range(len(classe_age)):
    classe_age_trad.append(i)

df['genercm'] = df['genercm'].replace(classe_age, classe_age_trad)
df['genercj'] = df['genercj'].replace(classe_age, classe_age_trad)
```

In [12]:
```
#Résumé des données
print(df[df.columns[2:4]].describe())
```

```
            genercm       genercj
count  2.121024e+06  2.121024e+06
mean   9.910599e+00  9.449836e+00
std    2.997679e+00  2.979134e+00
min    3.000000e+00  3.000000e+00
25%    7.000000e+00  7.000000e+00
50%    1.000000e+01  9.000000e+00
75%    1.200000e+01  1.200000e+01
max    2.000000e+01  2.000000e+01
```

In [13]:
```
#Nuage de points
df_classe_age = df.groupby(df.columns[2:4].tolist()).size().reset_index()
                                  .rename(columns={0:'Nbr'})
s = df_classe_age.Nbr
df_classe_age.plot.scatter('genercm', 'genercj', grid=False, figsize=(12, 8),
                           sharex=False, s=100, c=s, colormap='Reds')
xint = range(3, 21)
plt.xticks(xint)
plt.yticks(xint)
plt.savefig("Stats//sp_c_ages.png")
```

In [14]:
```
#Corrélation
print(df[df.columns[2:4]].corr())
```

```
           genercm   genercj
genercm   1.000000  0.952792
genercj   0.952792  1.000000
```

```
In [23]: #Analyse de la différence de groupe d'âges des partenaires
         plt.figure(figsize=(12, 8))
         plt.hist(abs(df.genercm-df.genercj), color='#5472AE', bins=25)
         plt.xlabel("Différence de groupe d'âges")
         plt.savefig('Stats//marriage_age_differences_true.png')

In [24]: from collections import Counter

         Counter(abs((df.genercm-df.genercj).values))

Out[24]: Counter({0: 1011447,
                  1: 893305,
                  2: 161681,
                  3: 37247,
                  4: 11483,
                  5: 3926,
                  6: 1328,
                  7: 405,
                  8: 137,
                  9: 40,
                  10: 20,
                  11: 4,
                  12: 1})
```

3 Analyse du diplôme

```
In [15]: #Transformation du groupe de diplômes en entier
         classe_diploma = ['IND', 'PRI', 'SEC', 'SUP']

         classe_diploma_trad = []
         for i in range(len(classe_diploma)):
             classe_diploma_trad.append(i)

         df['dipglcm'] = df['dipglcm'].replace(classe_diploma, classe_diploma_trad)
         df['dipglcj'] = df['dipglcj'].replace(classe_diploma, classe_diploma_trad)

In [16]: df_diploma = df.iloc[:, 8:10]

In [17]: #Retrait des valeurs nan
         df_diploma = df_diploma.dropna(how='any')

In [18]: #Résumé des données
         print(df_diploma.describe())

                dipglcm       dipglcj
count  1.985857e+06  1.985857e+06
mean   1.521760e+00  1.404373e+00
std    1.077806e+00  1.105399e+00
```

```
min    0.000000e+00  0.000000e+00
25%    0.000000e+00  0.000000e+00
50%    2.000000e+00  2.000000e+00
75%    2.000000e+00  2.000000e+00
max    3.000000e+00  3.000000e+00
```

```
In [19]: #Nuage de points
         df_diploma_s = df_diploma.groupby(df_diploma.columns[:].tolist()).size()
                                                    .reset_index()
                                                    .rename(columns={0:'Nbr'})
         s = df_diploma_s.Nbr
         df_diploma_s.plot.scatter('dipglcm', 'dipglcj', grid=False, sharex=False,
                                   c=s, s=100, colormap='Reds', figsize=(12, 8))
         xint = range(0, 4)
         plt.xticks(xint)
         plt.yticks(xint)
         plt.savefig("Stats//sp_diploma.png")
```

```
In [32]: #Corrélation
         print(df_diploma.corr())
```

```
          dipglcm   dipglcj
dipglcm  1.000000  0.559106
dipglcj  0.559106  1.000000
```

```
In [36]: #Analyse de la différence de groupe de diplômes entre les partenaires
         plt.figure(figsize=(12, 8))
         plt.hist(abs(df_diploma.dipglcm-df_diploma.dipglcj), color='#5472AE')
         plt.xlabel("Différence de groupe de diplômes")
         xint = range(0, 4)
         plt.xticks(xint)
         plt.savefig('Stats//marriage_diploma_differences_true.png')
```

```
In [41]: from collections import Counter

         Counter(abs((df_diploma.dipglcm-df_diploma.dipglcj).values))
```

```
Out[41]: Counter({0.0: 1156215, 1.0: 486365, 2.0: 292141, 3.0: 51136})
```

A.6 Préparation des données pour le premier modèle de choix

Preparation des donnees pour le premier modele de choix discrets

May 18, 2018

```
In [1]: #Packages utiles
        import pandas as pd
        import numpy as np
        from sklearn.model_selection import train_test_split
```

1 Chargement des données

```
In [2]: #Chargement des individus qui se sont mariés en 2000
        df_married = pd.read_csv('Donnees//data_CD1.csv', sep=' ')
```

```
In [3]: #Chargement des célibataires
        df_single = pd.read_csv('Donnees//data_single.csv', sep=' ')
```

2 Ajout du choix

```
In [4]: df_married['choice'] = np.ones(df_married.shape[0], dtype=int)
        df_single['choice'] = np.zeros(df_single.shape[0], dtype=int)
```

3 Assemblage des deux jeux de données

```
In [5]: df = pd.concat([df_married, df_single])
```

4 Ajout de la disponibilité pour Biogeme

```
In [7]: df['av'] = np.ones(df.shape[0], dtype=int)
```

5 Division des données en ensembles d'entrainement et de test

```
In [8]: df_train, df_test = train_test_split(df, train_size=0.7)
```

```
In [9]: df_train.to_csv('Donnees//data_biogeme1.txt', sep=' ', header=True, index=False)
```

```
In [10]: df_test.to_csv('Donnees//data_biosim1.txt', sep=' ', header=True, index=False)
```

A.7 Spécifications Python pour le premier modèle de choix

```
# -*- codage: utf-8 -*-
"""
Créé le 13 avril 2018
Auteur : Collot Alice
But : paramètres du modèle de choix discrets pour choisir le célibat
ou le mariage
Modèle : RUM - Logit
"""

#Packages utiles
from biogeme import *
from headers import *
from loglikelihood import *
from statistics import *


#Paramètres à estimer
ASC_married = Beta('ASC_married', 0, -100, 100, 0, 'Married cst')
ASC_single = Beta('ASC_single', 0, -100, 100, 1, 'Single cst')


B_sexe = Beta('B_sexe', 0, -100, 100, 0, 'Sexe')
B_diploma = Beta('B_diploma', 0, -100, 100, 0, 'Diploma')

B_age_G3 = Beta('B_age_G3', 0, -100, 100, 0, 'Age cat. 3')
B_age_G4 = Beta('B_age_G4', 0, -100, 100, 0, 'Age cat. 4')
B_age_G5 = Beta('B_age_G5', 0, -100, 100, 0, 'Age cat. 5')
B_age_G6 = Beta('B_age_G6', 0, -100, 100, 0, 'Age cat. 6')
B_age_G7 = Beta('B_age_G7', 0, -100, 100, 0, 'Age cat. 7')
B_age_G8 = Beta('B_age_G8', 0, -100, 100, 0, 'Age cat. 8')
B_age_G9 = Beta('B_age_G9', 0, -100, 100, 0, 'Age cat. 9')
B_age_G10 = Beta('B_age_G10', 0, -100, 100, 0, 'Age cat. 10')
B_age_G11 = Beta('B_age_G11', 0, -100, 100, 0, 'Age cat. 11')
B_age_G12 = Beta('B_age_G12', 0, -100, 100, 0, 'Age cat. 12')
B_age_G13 = Beta('B_age_G13', 0, -100, 100, 0, 'Age cat. 13')
B_age_G14 = Beta('B_age_G14', 0, -100, 100, 0, 'Age cat. 14')
B_age_G15 = Beta('B_age_G15', 0, -100, 100, 0, 'Age cat. 15')
B_age_G16 = Beta('B_age_G16', 0, -100, 100, 0, 'Age cat. 16')
B_age_G17 = Beta('B_age_G17', 0, -100, 100, 0, 'Age cat. 17')
B_age_G18 = Beta('B_age_G18', 0, -100, 100, 0, 'Age cat. 18')
B_age_G19 = Beta('B_age_G19', 0, -100, 100, 0, 'Age cat. 19')
B_age_G20 = Beta('B_age_G10', 0, -100, 100, 1, 'Age cat. 20')


#Fonctions d'utilité
GAge3 = DefineVariable('GAge3', grage == 3)
GAge4 = DefineVariable('GAge4', grage == 4)
GAge5 = DefineVariable('GAge5', grage == 5)
GAge6 = DefineVariable('GAge6', grage == 6)
GAge7 = DefineVariable('GAge7', grage == 7)
GAge8 = DefineVariable('GAge8', grage == 8)
GAge9 = DefineVariable('GAge9', grage == 9)
GAge10 = DefineVariable('GAge10', grage == 10)
GAge11 = DefineVariable('GAge11', grage == 11)
```

```
GAge12 = DefineVariable('GAge12', grage == 12)
GAge13 = DefineVariable('GAge13', grage == 13)
GAge14 = DefineVariable('GAge14', grage == 14)
GAge15 = DefineVariable('GAge15', grage == 15)
GAge16 = DefineVariable('GAge16', grage == 16)
GAge17 = DefineVariable('GAge17', grage == 17)
GAge18 = DefineVariable('GAge18', grage == 18)
GAge19 = DefineVariable('GAge19', grage == 19)
GAge20 = DefineVariable('GAge20', grage == 20)


v1 = ASC_married * one + B_sexe * sexe + B_diploma * diploma
+ B_age_G3 * GAge3 + B_age_G4 * GAge4 + B_age_G5 * GAge5
+ B_age_G6 * GAge6 + B_age_G7 * GAge7 + B_age_G8 * GAge8
+ B_age_G9 * GAge9 + B_age_G10 * GAge10 + B_age_G11 * GAge11
+ B_age_G12 * GAge12 + B_age_G13 * GAge13 + B_age_G14 * GAge14
+ B_age_G15 * GAge15 + B_age_G16 * GAge16 + B_age_G17 * GAge17
+ B_age_G18 * GAge18 + B_age_G19 * GAge19 + B_age_G20 * GAge20

v2 = ASC_single * one


#Association des fonctions d'utilité avec les alternatives
V = {1: v1, 0: v2}

#Association des conditions de disponibilités avec les alternatives
AV = {1: av,
      0: av}

#Modèle de choix est un logit
logprob = bioLogLogit(V, AV, choice)

#Itérateur sur les données
rowIterator('obsIter')

#Fonction de vraisemblance pour l'estimation
BIOGEME_OBJECT.ESTIMATE = Sum(logprob, 'obsIter')


#Statistiques
nullLoglikelihood(AV, 'obsIter')
choiceSet = [0, 1]
cteLoglikelihood(choiceSet, choice, 'obsIter')


BIOGEME_OBJECT.PARAMETERS['optimizationAlgorithm'] = "BIO"

BIOGEME_OBJECT.FORMULAS['Married utility'] = v1
BIOGEME_OBJECT.FORMULAS['Single utility'] = v2
```

A.8 Vérification du premier modèle de choix discrets

Verification du premier modele de choix discrets sur la decision de se marier une annee particuliere

May 24, 2018

In [1]:
```
#Packages utiles
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import itertools
import time
from random import random
from math import exp
from sklearn.metrics import confusion_matrix
```

In [2]:
```
#Taille du texte pour les graphes
plt.rc('font', size=18)
```

1 Chargement des données

In [3]:
```
df_test = pd.read_csv('Donnees//data_biosim1.txt', sep=' ')
```

In [4]:
```
df_test['GAge3'] = (df_test.grage == 3)*1
df_test['GAge4'] = (df_test.grage == 4)*1
df_test['GAge5'] = (df_test.grage == 5)*1
df_test['GAge6'] = (df_test.grage == 6)*1
df_test['GAge7'] = (df_test.grage == 7)*1
df_test['GAge8'] = (df_test.grage == 8)*1
df_test['GAge9'] = (df_test.grage == 9)*1
df_test['GAge10'] = (df_test.grage == 10)*1
df_test['GAge11'] = (df_test.grage == 11)*1
df_test['GAge12'] = (df_test.grage == 12)*1
df_test['GAge13'] = (df_test.grage == 13)*1
df_test['GAge14'] = (df_test.grage == 14)*1
df_test['GAge15'] = (df_test.grage == 15)*1
df_test['GAge16'] = (df_test.grage == 16)*1
df_test['GAge17'] = (df_test.grage == 17)*1
df_test['GAge18'] = (df_test.grage == 18)*1
df_test['GAge19'] = (df_test.grage == 19)*1
df_test['GAge20'] = (df_test.grage == 20)*1
```

2 Chargement du modèle de choix discrets

In [5]:
```
#Chargement des valeurs de paramètres
parameter_BIO = [-10.8583, 0, 0.119786, -1.45089, 6.05381, 9.20658, 9.75218,
                 9.04087, 8.26914, 7.71428, 7.34039, 6.97391, 6.22731, 5.5735,
                 4.71758, 4.05904, 3.69685, 2.95563, 2.35302, 2.19859, 0, 0]

ASC_married = parameter_BIO[0]
ASC_single = parameter_BIO[1]

B_sexe = parameter_BIO[2]
B_diploma = parameter_BIO[3]
B_age_G3 = parameter_BIO[4]
B_age_G4 = parameter_BIO[5]
B_age_G5 = parameter_BIO[6]
B_age_G6 = parameter_BIO[7]
B_age_G7 = parameter_BIO[8]
B_age_G8 = parameter_BIO[9]
B_age_G9 = parameter_BIO[10]
B_age_G10 = parameter_BIO[11]
B_age_G11 = parameter_BIO[12]
B_age_G12 = parameter_BIO[13]
B_age_G13 = parameter_BIO[14]
B_age_G14 = parameter_BIO[15]
B_age_G15 = parameter_BIO[16]
B_age_G16 = parameter_BIO[17]
B_age_G17 = parameter_BIO[18]
B_age_G18 = parameter_BIO[19]
B_age_G19 = parameter_BIO[20]
B_age_G20 = parameter_BIO[21]
```

3 Vérification du modèle

In [ ]:
```
start_time = time.clock()
time_i = []

nbr_marriage = np.zeros(10)

for i in range(10):

    cd_choice = []

    for index, person in df_test.iterrows():

        nbr = random()

        cd = ASC_married + B_sexe * person.sexe + B_diploma * person.diploma +
```

```
                B_age_G3 * person.GAge3 + B_age_G4 * person.GAge4 +
                B_age_G5 * person.GAge5 + B_age_G6 * person.GAge6 +
                B_age_G7 * person.GAge7 + B_age_G8 * person.GAge8 +
                B_age_G9 * person.GAge9 + B_age_G10 * person.GAge10 +
                B_age_G11 * person.GAge11 + B_age_G12 * person.GAge12 +
                B_age_G13 * person.GAge13 + B_age_G14 * person.GAge14 +
                B_age_G15 * person.GAge15 + B_age_G16 * person.GAge16 +
                B_age_G17 * person.GAge17 + B_age_G18 * person.GAge18 +
                B_age_G19 * person.GAge19 + B_age_G20 * person.GAge20

                proba = exp(cd)/(exp(cd)+exp(ASC_single))

                if(nbr<proba):
                    cd_choice.append(1)
                    nbr_marriage[i] += 1
                else:
                    cd_choice.append(0)

            time_i.append(time.clock()-start_time)
```

In [36]:
```
#Affichage du temps d'exécution par simulation
plt.figure(figsize=(10, 8))

x = [i for i in range(10)]

plt.plot(x, time_i, '-', linewidth=2)
plt.grid(True)
plt.xlabel('Numéro de simulation')
plt.ylabel("Temps d'exécution (s)")
plt.savefig('Stats//line_time_spend_M1.png')
```

In [ ]:
```
#Affichage du nombre de personnes dans le marché par simulation
plt.figure(figsize=(14, 8))

x = [i for i in range(10)]

plt.plot(x, nbr_marriage, '-', linewidth=2)
plt.grid(True)
plt.xlabel('Numéro de simulation')
plt.ylabel('Nombre de personnes')
plt.savefig('Stats//line_nbr_mar_M1.png')
```

4 Affichage des résultats

In [17]:
```
df_test['pred_choice'] = cd_choice
```

In [18]:
```
df_H = df_test[df_test.sexe==1][['choice', 'grage', 'pred_choice']]
df_F = df_test[df_test.sexe==2][['choice', 'grage', 'pred_choice']]
```

```
In [19]: df_H_true = pd.crosstab(df_H.grage, df_H.choice)
         df_H_pred = pd.crosstab(df_H.grage, df_H.pred_choice)

         df_F_true = pd.crosstab(df_F.grage, df_F.choice)
         df_F_pred = pd.crosstab(df_F.grage, df_F.pred_choice)
```

4.1 Affichage du nombre d'individus dans le marché

```
In [ ]: plt.figure(figsize=(14, 8))

        x = [i for i in range(10)]
        z = [df_test['choice'].value_counts()[1] for i in range(10)]

        plt.plot(x, nbr_marriage, '-', linewidth=2)

        plt.plot(x, z, '-', linewidth=2)

        plt.grid(True)
        plt.legend(['Nombre prédit', 'Nombre réel'])
        plt.xlabel('Numéro de simulation')
        plt.ylabel('Nombre de personnes')
        plt.savefig('Stats//line_nbr_marriage_M1.png')
        plt.show()
```

4.2 Matrice de confusion

```
In [23]: cf_matrix = confusion_matrix(df_test['choice'], df_test['pred_choice'])
```

```
In [24]: def plot_confusion_matrix(cm, classes, normalize=False,
                                   title='Confusion matrix', cmap=plt.cm.Blues):
             """
             Cette fonction affiche et dessine une matrice de confusion.
             La normalisation peut être appliquée avec `normalize=True`.
             Code provenant d'un exemple sklearn.
             """
             if normalize:
                 cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
                 print("Normalized confusion matrix")
             else:
                 print('Confusion matrix, without normalization')

             print(cm)

             plt.imshow(cm, interpolation='nearest', cmap=cmap)
             plt.title(title)
             plt.colorbar()
             tick_marks = np.arange(len(classes))
             plt.xticks(tick_marks, classes, rotation=45)
```

```
            plt.yticks(tick_marks, classes)

            fmt = '.2f' if normalize else 'd'
            thresh = cm.max() / 2.
            for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
                plt.text(j, i, format(cm[i, j], fmt),
                         horizontalalignment="center",
                         color="white" if cm[i, j] > thresh else "black")

            plt.tight_layout()
            plt.ylabel('Label réel')
            plt.xlabel('Label prédit')
```

```
In [32]: plt.figure()
         plot_confusion_matrix(cf_matrix, classes=['Célibataire', 'Marié'])
         plt.savefig("Stats//Conf_matrix_M1.png")
```

```
Confusion matrix, without normalization
[[2008683   19283]
 [  19028    3374]]
```

```
In [33]: plt.figure()
         plot_confusion_matrix(cf_matrix, classes=['Célibataire', 'Marié'], normalize=True)
         plt.savefig("Stats//Conf_matrix_N_M1.png")
```

```
Normalized confusion matrix
[[ 0.99049146  0.00950854]
 [ 0.84938845  0.15061155]]
```

4.3 Graphe en bâtonnets pour le groupe d'âges par sexe

```
In [ ]: plt.figure(figsize=(12, 8))
        ax = plt.subplot(111)
        ax.bar(df_H_true.index-0.1, df_H_true[1], width=0.2, color='#5472AE', align='center')
        ax.bar(df_H_pred.index+0.1, df_H_pred[1], width=0.2, color='#FEA347', align='center')
        ax.legend(['Choix réel', 'Choix prédit'])
        plt.xlabel("Catégorie d'âges")
        plt.ylabel("Nombre d'hommes")
        xint = range(3, 20)
        plt.xticks(xint)
        plt.savefig('Stats//barchart_m_M1.png')
```

```
In [60]: print('Différence entre les choix réel et prédit des hommes:',
                sum(abs(df_H_true[1]-df_H_pred[1]).values))
```

```
Différence entre les choix réel et prédit des hommes : 1474
```

```
In [ ]: plt.figure(figsize=(12, 8))
        ax = plt.subplot(111)
        ax.bar(df_F_true.index-0.1, df_F_true[1], width=0.2, color='#5472AE', align='center')
        ax.bar(df_F_pred.index+0.1, df_F_pred[1], width=0.2, color='#FEA347', align='center')
        ax.legend(['Choix réel', 'Choix prédit'])
        plt.xlabel("Catégorie d'âges")
        plt.ylabel('Nombre de femmes')
        xint = range(3, 20)
        plt.xticks(xint)
        plt.savefig('Stats//barchart_w_M1.png')

In [61]: print('Différence entre les choix réel et prédit des femmes:',
               sum(abs(df_F_true[1]-df_F_pred[1]).values))

Différence entre les choix réel et prédit des femmes : 1647
```

4.4 Graphe en bâtonnets pour le diplôme par sexe

```
In [62]: df_H = df_test[df_test.sexe==1][['choice', 'diploma', 'pred_choice']]
         df_F = df_test[df_test.sexe==2][['choice', 'diploma', 'pred_choice']]

         df_H_true = pd.crosstab(df_H.diploma, df_H.choice)
         df_H_pred = pd.crosstab(df_H.diploma, df_H.pred_choice)

         df_F_true = pd.crosstab(df_F.diploma, df_F.choice)
         df_F_pred = pd.crosstab(df_F.diploma, df_F.pred_choice)

In [64]: plt.figure(figsize=(12, 8))
         ax = plt.subplot(111)
         ax.bar(df_H_true.index-0.1, df_H_true[1], width=0.2, color='#5472AE', align='center')
         ax.bar(df_H_pred.index+0.1, df_H_pred[1], width=0.2, color='#FEA347', align='center')
         ax.legend(['Choix réel', 'Choix prédit'])
         plt.xlabel("Catégorie d'âges")
         plt.ylabel("Nombre d'hommes")
         xint = range(0, 4)
         plt.xticks(xint)
         plt.savefig('Stats//barchart_m_M1_diploma.png')

In [65]: print('Différence entre les choix réel et prédit des hommes:',
               sum(abs(df_H_true[1]-df_H_pred[1]).values))

Différence entre les choix réel et prédit des hommes: 1154


In [69]: plt.figure(figsize=(12, 8))
         ax = plt.subplot(111)
         ax.bar(df_F_true.index-0.1, df_F_true[1], width=0.2, color='#5472AE', align='center')
         ax.bar(df_F_pred.index+0.1, df_F_pred[1], width=0.2, color='#FEA347', align='center')
```

```
        ax.legend(['Choix réel', 'Choix prédit'])
        plt.xlabel("Catégorie d'âges")
        plt.ylabel('Nombre de femmes')
        xint = range(0, 4)
        plt.xticks(xint)
        plt.savefig('Stats//barchart_w_M1_diploma.png')
```

```
In [70]: print('Différence entre les choix réel et prédit des femmes:',
              sum(abs(df_F_true[1]-df_F_pred[1]).values))
```

```
Différence entre les choix réel et prédit des femmes: 2547
```

A.9 Implémentation du premier réseau de neurones

Reseau de neurones pour construire le marche des mariages

May 24, 2018

In [1]:
```
#Packages utiles
import itertools
import time
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from random import random
from sklearn.model_selection import GridSearchCV
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import confusion_matrix
```

In [2]:
```
#Taille du texte pour les graphes
plt.rc('font', size=18)
```

1 Chargement des données

In [3]:
```
df_train = pd.read_csv('Donnees//data_biogeme1.txt', sep=' ')
```

In [4]:
```
X_train = df_train[['diploma', 'grage', 'sexe']]
y_train = df_train['choice']
```

In [5]:
```
df_test = pd.read_csv('Donnees//data_biosim1.txt', sep=' ')
```

In [6]:
```
X_test = df_test[['diploma', 'grage', 'sexe']]
y_test = df_test['choice']
```

2 Réseau de neurones

In [ ]:
```
start_time = time.clock()

hidden_layer = [(26,), (27,), (28,)]

parameters = {'hidden_layer_sizes': hidden_layer}

aNN = GridSearchCV(MLPClassifier(solver='adam', activation='relu'),
                   param_grid=parameters, cv=10)
aNN.fit(X_train, y_train)
```

```
time_calibr = time.clock()-start_time

proba_aNN = aNN.predict_proba(X_test)
```

3 Vérification du réseau de neurones

```
In [ ]: start_time = time.clock()

        nbr_marriage = np.zeros(10)

        for i in range(10):

            cd_choice = []

            for index, person in df_test.iterrows():

                nbr = random()

                proba = proba_aNN[index][1]

                if(nbr<proba):
                    cd_choice.append(1)
                    nbr_marriage[i] += 1
                else:
                    cd_choice.append(0)

        time_pred = time.clock()-start_time
```

```
In [30]: df_test['pred_choice'] = cd_choice

         df_H = df_test[df_test.sexe==1][['choice', 'grage', 'diploma', 'pred_choice']]
         df_F = df_test[df_test.sexe==2][['choice', 'grage', 'diploma', 'pred_choice']]

         df_H_true = pd.crosstab(df_H.grage, df_H.choice)
         df_H_pred = pd.crosstab(df_H.grage, df_H.pred_choice)

         df_F_true = pd.crosstab(df_F.grage, df_F.choice)
         df_F_pred = pd.crosstab(df_F.grage, df_F.pred_choice)
```

3.1 Nombre de personnes dans le marché pour chaque simulation

```
In [72]: plt.figure(figsize=(10, 8))

         x = [i for i in range(10)]
         z = [df_test['choice'].value_counts()[1] for i in range(10)]

         plt.plot(x, nbr_marriage, '-', linewidth=2)
```

```
plt.plot(x, z, '-', linewidth=2)

plt.grid(True)
plt.legend(['Nombre prédit', 'Nombre réel'])
plt.xlabel('Numéro de simulation')
plt.ylabel('Nombre de personnes')
plt.savefig('Stats//line_nbr_marriage_aNN_relu.png')
```

3.2 Matrice de confusion

```
In [23]: cf_matrix = confusion_matrix(df_test['choice'], df_test['pred_choice'])
```

```
In [37]: def plot_confusion_matrix(cm, classes, normalize=False,
                                   title='Confusion matrix', cmap=plt.cm.Blues):
             """
             Cette fonction affiche et dessine une matrice de confusion.
             La normalisation peut être appliquée avec 'normalize=True'.
             Code provenant d'un exe.mple sklearn
             """
             if normalize:
                 cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
                 print("Normalized confusion matrix")
             else:
                 print('Confusion matrix, without normalization')

             print(cm)

             plt.imshow(cm, interpolation='nearest', cmap=cmap)
             plt.title(title)
             plt.colorbar()
             tick_marks = np.arange(len(classes))
             plt.xticks(tick_marks, classes, rotation=45)
             plt.yticks(tick_marks, classes)

             fmt = '.2f' if normalize else 'd'
             thresh = cm.max() / 2.
             for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
                 plt.text(j, i, format(cm[i, j], fmt),
                          horizontalalignment="center",
                          color="white" if cm[i, j] > thresh else "black")

             plt.tight_layout()
             plt.ylabel('Label réel')
             plt.xlabel('Label prédit')
```

```
In [49]: plt.figure()
         plot_confusion_matrix(cf_matrix, classes=['Célibataire', 'Marié'])
         plt.savefig("Stats//Conf_matrix_ANN_relu.png")
```

```
Confusion matrix, without normalization
[[2009701   18265]
 [  18075    4327]]
```

```
In [51]: plt.figure()
         plot_confusion_matrix(cf_matrix, classes=['Célibataire', 'Marié'], normalize=True)
         plt.savefig("Stats//Conf_matrix_ANN_relu_N.png")
```

```
Normalized confusion matrix
[[ 0.99099344  0.00900656]
 [ 0.8068476   0.1931524 ]]
```

3.3 Graphe en bâtonnets pour le groupe d'âges par sexe

```
In [55]: plt.figure(figsize=(12, 8))
         ax = plt.subplot(111)
         ax.bar(df_H_true.index-0.1, df_H_true[1], width=0.2, color='#5472AE', align='center')
         ax.bar(df_H_pred.index+0.1, df_H_pred[1], width=0.2, color='#FEA347', align='center')
         ax.legend(['Choix réel', 'Choix prédit'])
         plt.xlabel("Catégorie d'âges")
         plt.ylabel("Nombre d'hommes")
         xint = range(3, 20)
         plt.xticks(xint)
         plt.savefig('Stats//barchart_m_ANN_grage_relu.png')
```

```
In [59]: print('Différence entre les choix réel et prédit des hommes:',
               sum(abs(df_H_true[1]-df_H_pred[1]).values))
```

```
Différence entre les choix réel et prédit des hommes: 1046
```

```
In [58]: plt.figure(figsize=(12, 8))
         ax = plt.subplot(111)
         ax.bar(df_F_true.index-0.1, df_F_true[1], width=0.2, color='#5472AE', align='center')
         ax.bar(df_F_pred.index+0.1, df_F_pred[1], width=0.2, color='#FEA347', align='center')
         ax.legend(['Choix réel', 'Choix prédit'])
         plt.xlabel("Catégorie d'âges")
         plt.ylabel("Nombre de femmes")
         xint = range(3, 20)
         plt.xticks(xint)
         plt.savefig('Stats//barchart_F_ANN_grage_relu.png')
```

```
In [60]: print('Différence entre les choix réel et prédit des femmes:',
               sum(abs(df_F_true[1]-df_F_pred[1]).values))
```

```
Différence entre les choix réel et prédit des femmes: 788
```

3.4 Graphe en bâtonnets pour le diplôme par sexe

```
In [61]: df_H_true = pd.crosstab(df_H.diploma, df_H.choice)
         df_H_pred = pd.crosstab(df_H.diploma, df_H.pred_choice)

         df_F_true = pd.crosstab(df_F.diploma, df_F.choice)
         df_F_pred = pd.crosstab(df_F.diploma, df_F.pred_choice)
```

```
In [71]: plt.figure(figsize=(12, 8))
         ax = plt.subplot(111)
         ax.bar(df_H_true.index-0.1, df_H_true[1], width=0.2, color='#5472AE', align='center')
         ax.bar(df_H_pred.index+0.1, df_H_pred[1], width=0.2, color='#FEA347', align='center')
         ax.legend(['Choix réel', 'Choix prédit'])
         plt.xlabel("Catégorie d'âges")
         plt.ylabel("Nombre d'hommes")
         xint = range(0, 4)
         plt.xticks(xint)
         plt.savefig('Stats//barchart_m_ANN_diploma_relu.png')
```

```
In [66]: print('Différence entre les choix réel et prédit des hommes:',
               sum(abs(df_H_true[1]-df_H_pred[1]).values))
```

```
Différence entre les choix réel et prédit des hommes: 754
```

```
In [70]: plt.figure(figsize=(12, 8))
         ax = plt.subplot(111)
         ax.bar(df_F_true.index-0.1, df_F_true[1], width=0.2, color='#5472AE', align='center')
         ax.bar(df_F_pred.index+0.1, df_F_pred[1], width=0.2, color='#FEA347', align='center')
         ax.legend(['Choix réel', 'Choix prédit'])
         plt.xlabel("Catégorie d'âges")
         plt.ylabel("Nombre de femmes")
         xint = range(0, 4)
         plt.xticks(xint)
         plt.savefig('Stats//barchart_F_ANN_diploma_relu.png')
```

```
In [69]: print('Différence entre les choix réel et prédit des femmes:',
               sum(abs(df_F_true[1]-df_F_pred[1]).values))
```

```
Différence entre les choix réel et prédit des femmes: 530
```

A.10 Préparation des données pour le second modèle de choix discrets

Preparation des donnees pour les modeles definissant les preferences des hommes

May 25, 2018

In [1]:
```
#Packages utiles
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
```

1 Chargement et transformation des données relatives aux individus mariés

In [2]:
```
df_married = pd.read_csv("Donnees//suivicouples.txt", sep="\t")
```

In [5]:
```
list_of_values = [year for year in range(1991, 2001)]

df_married = df_married[df_married['acheciv'].isin(list_of_values)]

#Index corrects
df_married = df_married.reset_index()
```

In [6]:
```
df_married = df_married[['sexecm', 'genercm', 'genercj', 'dipglcm', 'dipglcj']]
```

In [7]:
```
#Le sexe du conjoint est l'opposé de celui du chef de ménage
df_married['sexecj'] = df_married.sexecm
df_married['sexecj'] = df_married['sexecj'].replace([1, 2], [2, 1])

#Ajout de la disponibilité pour Biogeme
df_married['av'] = np.ones(df_married.shape[0], dtype=int)
```

In [8]:
```
#Remplacement des valeurs nan pour éviter les erreurs
df_married = df_married.replace(np.nan, 'IND', regex=True)
```

In [9]:
```
#Transformation des classes de diplômes en entier
classe_diploma = ['IND', 'PRI', 'SEC', 'SUP']

classe_diploma_trad = []
for i in range(len(classe_diploma)):
    classe_diploma_trad.append(i)
```

```
df_married['dipglcm'] = df_married['dipglcm'].replace(classe_diploma,
                                                      classe_diploma_trad)
df_married['dipglcj'] = df_married['dipglcj'].replace(classe_diploma,
                                                      classe_diploma_trad)
```

```
In [10]: #Transformation des classes d'âges en entier
         classe_age = ['00-05', '05-10', '10-15', '15-20', '20-25', '25-30', '30-35',
                       '35-40', '40-45', '45-50', '50-55', '55-60', '60-65', '65-70',
                       '70-75', '75-80', '80-85', '85-90', '90-95', '95-100', '100+']

         classe_age_trad = []
         for i in range(len(classe_age)):
             classe_age_trad.append(i)

         df_married['genercm'] = df_married['genercm'].replace(classe_age, classe_age_trad)
         df_married['genercj'] = df_married['genercj'].replace(classe_age, classe_age_trad)
```

2 Séparation des données en ensembles d'entrainement et de test

```
In [13]: df_train, df_test = train_test_split(df_married, train_size=0.7)
```

3 Séparation des données par sexe

```
In [16]: df_train_cm = df_train[['sexecm', 'genercm', 'genercj', 'dipglcm', 'dipglcj', 'av']]
         df_train_cm.columns = ['sexe', 'grage', 'grage_C', 'diploma', 'diploma_C', 'av']

         df_train_cj = df_train[['sexecj', 'genercj', 'genercm', 'dipglcj', 'dipglcm', 'av']]
         df_train_cj.columns = ['sexe', 'grage', 'grage_C', 'diploma', 'diploma_C', 'av']


         df_test_cm = df_test[['sexecm', 'genercm', 'genercj', 'dipglcm', 'dipglcj', 'av']]
         df_test_cm.columns = ['sexe', 'grage', 'grage_C', 'diploma', 'diploma_C', 'av']

         df_test_cj = df_test[['sexecj', 'genercj', 'genercm', 'dipglcj', 'dipglcm', 'av']]
         df_test_cj.columns = ['sexe', 'grage', 'grage_C', 'diploma', 'diploma_C', 'av']
```

```
In [17]: #Fusion des df
         df_train_men = pd.concat([df_train_cm[df_train_cm.sexe==1],
                                   df_train_cj[df_train_cj.sexe==1]])
         df_train_women = pd.concat([df_train_cm[df_train_cm.sexe==2],
                                     df_train_cj[df_train_cj.sexe==2]])

         df_test_men = pd.concat([df_test_cm[df_test_cm.sexe==1],
                                  df_test_cj[df_test_cj.sexe==1]])
         df_test_women = pd.concat([df_test_cm[df_test_cm.sexe==2],
                                    df_test_cj[df_test_cj.sexe==2]])
```

4 Sauvegarde des données

In [18]:
```
df_train_men.to_csv('Donnees//data_train_men.txt', sep=' ', header=True,
                    index=False)
df_train_women.to_csv('Donnees//data_train_women.txt', sep=' ', header=True,
                      index=False)

df_test_men.to_csv('Donnees//data_test_men.txt', sep=' ', header=True,
                   index=False)
df_test_women.to_csv('Donnees//data_test_women.txt', sep=' ', header=True,
                     index=False)
```

A.11 Spécifications Python pour le second modèle de choix

```
# -*- codage: utf-8 -*-
"""
Créé le 22 avril 2018
Auteur : Collot Alice
But : paramètres du modèle de choix pour les préférences des hommes
Model : RUM - Logit
"""

#Packages utiles
from biogeme import *
from headers import *
from loglikelihood import *
from statistics import *


#Paramètres à estimer
ASC_V3 = Beta('ASC_V3', 0, -100, 100, 0, 'Age cat. 3 cst')
ASC_V4 = Beta('ASC_V4', 0, -100, 100, 0, 'Age cat. 4 cst')
ASC_V5 = Beta('ASC_V5', 0, -100, 100, 0, 'Age cat. 5 cst')
ASC_V6 = Beta('ASC_V6', 0, -100, 100, 0, 'Age cat. 6 cst')
ASC_V7 = Beta('ASC_V7', 0, -100, 100, 1, 'Age cat. 7 cst')
ASC_V8 = Beta('ASC_V8', 0, -100, 100, 0, 'Age cat. 8 cst')
ASC_V9 = Beta('ASC_V9', 0, -100, 100, 0, 'Age cat. 9 cst')
ASC_V10 = Beta('ASC_V10', 0, -100, 100, 0, 'Age cat. 10 cst')
ASC_V11 = Beta('ASC_V11', 0, -100, 100, 0, 'Age cat. 11 cst')
ASC_V12 = Beta('ASC_V12', 0, -100, 100, 0, 'Age cat. 12 cst')
ASC_V13 = Beta('ASC_V13', 0, -100, 100, 0, 'Age cat. 13 cst')
ASC_V14 = Beta('ASC_V14', 0, -100, 100, 0, 'Age cat. 14 cst')
ASC_V15 = Beta('ASC_V15', 0, -100, 100, 0, 'Age cat. 15 cst')
ASC_V16 = Beta('ASC_V16', 0, -100, 100, 0, 'Age cat. 16 cst')
ASC_V17 = Beta('ASC_V17', 0, -100, 100, 0, 'Age cat. 17 cst')
ASC_V18 = Beta('ASC_V18', 0, -100, 100, 0, 'Age cat. 18 cst')
ASC_V19 = Beta('ASC_V19', 0, -100, 100, 0, 'Age cat. 19 cst')
ASC_V20 = Beta('ASC_V20', 0, -100, 100, 1, 'Age cat. 20 cst')


B_age_G3 = Beta('B_age_G3', 0, -100, 100, 0, 'Age cat. 3')
B_age_G4 = Beta('B_age_G4', 0, -100, 100, 0, 'Age cat. 4')
B_age_G5 = Beta('B_age_G5', 0, -100, 100, 0, 'Age cat. 5')
B_age_G6 = Beta('B_age_G6', 0, -100, 100, 0, 'Age cat. 6')
B_age_G7 = Beta('B_age_G7', 0, -100, 100, 0, 'Age cat. 7')
B_age_G8 = Beta('B_age_G8', 0, -100, 100, 0, 'Age cat. 8')
B_age_G9 = Beta('B_age_G9', 0, -100, 100, 0, 'Age cat. 9')
B_age_G10 = Beta('B_age_G10', 0, -100, 100, 0, 'Age cat. 10')
B_age_G11 = Beta('B_age_G11', 0, -100, 100, 0, 'Age cat. 11')
B_age_G12 = Beta('B_age_G12', 0, -100, 100, 0, 'Age cat. 12')
B_age_G13 = Beta('B_age_G13', 0, -100, 100, 0, 'Age cat. 13')
B_age_G14 = Beta('B_age_G14', 0, -100, 100, 0, 'Age cat. 14')
B_age_G15 = Beta('B_age_G15', 0, -100, 100, 0, 'Age cat. 15')
B_age_G16 = Beta('B_age_G16', 0, -100, 100, 0, 'Age cat. 16')
B_age_G17 = Beta('B_age_G17', 0, -100, 100, 0, 'Age cat. 17')
B_age_G18 = Beta('B_age_G18', 0, -100, 100, 0, 'Age cat. 18')
B_age_G19 = Beta('B_age_G19', 0, -100, 100, 0, 'Age cat. 19')
B_age_G20 = Beta('B_age_G10', 0, -100, 100, 1, 'Age cat. 20')
```

```
55
56
57
58 #Fonctions d'utilité
59 V3 = ASC_V3 * one + B_age_G3 * grage
60 V4 = ASC_V4 * one + B_age_G4 * grage
61 V5 = ASC_V5 * one + B_age_G5 * grage
62 V6 = ASC_V6 * one + B_age_G6 * grage
63 V7 = ASC_V7 * one + B_age_G7 * grage
64 V8 = ASC_V8 * one + B_age_G8 * grage
65 V9 = ASC_V9 * one + B_age_G9 * grage
66 V10 = ASC_V10 * one + B_age_G10 * grage
67 V11 = ASC_V11 * one + B_age_G11 * grage
68 V12 = ASC_V12 * one + B_age_G12 * grage
69 V13 = ASC_V13 * one + B_age_G13 * grage
70 V14 = ASC_V14 * one + B_age_G14 * grage
71 V15 = ASC_V15 * one + B_age_G15 * grage
72 V16 = ASC_V16 * one + B_age_G16 * grage
73 V17 = ASC_V17 * one + B_age_G17 * grage
74 V18 = ASC_V18 * one + B_age_G18 * grage
75 V19 = ASC_V19 * one + B_age_G19 * grage
76 V20 = ASC_V20 * one
77
78
79 #Association des fonctions d'utilité avec les alternatives
80 V = {3: V3, 4: V4, 5: V5, 6: V6, 7: V7, 8: V8, 9: V9, 10: V10, 11: V11,
81      12: V12, 13: V13, 14: V14, 15: V15, 16: V16, 17: V17, 18: V18,
82      19: V19, 20: V20}
83
84 #Association des disponibilités avec les alternatives
85 AV = {3: av,
86       4: av,
87       5: av,
88       6: av,
89       7: av,
90       8: av,
91       9: av,
92       10: av,
93       11: av,
94       12: av,
95       13: av,
96       14: av,
97       15: av,
98       16: av,
99       17: av,
100      18: av,
101      19: av,
102      20: av}
103
104 #Le modèle de choix est un logit
105 logprob = bioLogLogit(V, AV, grage_C)
106
107 #Définition d'un itérateur sur les données
108 rowIterator('obsIter')
109
110 #Définition de la fonction de vraisemblance pour l'estimation
```

```
111 BIOGEME_OBJECT.ESTIMATE = Sum(logprob,'obsIter')
112
113
114 #Statistiques
115 nullLoglikelihood(AV, 'obsIter')
116 choiceSet = [3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18,
117              19, 20]
118 cteLoglikelihood(choiceSet, grage_C,'obsIter')
119
120
121 BIOGEME_OBJECT.PARAMETERS['optimizationAlgorithm'] = "BIO"
122
123 BIOGEME_OBJECT.FORMULAS['V3 utility'] = V3
124 BIOGEME_OBJECT.FORMULAS['V4 utility'] = V4
125 BIOGEME_OBJECT.FORMULAS['V5 utility'] = V5
126 BIOGEME_OBJECT.FORMULAS['V6 utility'] = V6
127 BIOGEME_OBJECT.FORMULAS['V7 utility'] = V7
128 BIOGEME_OBJECT.FORMULAS['V8 utility'] = V8
129 BIOGEME_OBJECT.FORMULAS['V9 utility'] = V9
130 BIOGEME_OBJECT.FORMULAS['V10 utility'] = V10
131 BIOGEME_OBJECT.FORMULAS['V11 utility'] = V11
132 BIOGEME_OBJECT.FORMULAS['V12 utility'] = V12
133 BIOGEME_OBJECT.FORMULAS['V13 utility'] = V13
134 BIOGEME_OBJECT.FORMULAS['V14 utility'] = V14
135 BIOGEME_OBJECT.FORMULAS['V15 utility'] = V15
136 BIOGEME_OBJECT.FORMULAS['V16 utility'] = V16
137 BIOGEME_OBJECT.FORMULAS['V17 utility'] = V17
138 BIOGEME_OBJECT.FORMULAS['V18 utility'] = V18
139 BIOGEME_OBJECT.FORMULAS['V19 utility'] = V19
140 BIOGEME_OBJECT.FORMULAS['V20 utility'] = V20
```

A.12 Vérification du second modèle de choix

Verification du modele de choix discrets determinant les preferences des hommes

May 25, 2018

```
In [ ]: #Packages utiles
        import pandas as pd
        import matplotlib.pyplot as plt
        import numpy as np
        import itertools
        import time
        from random import random
        from math import exp
        from pylab import setp
```

```
In [ ]: #Taille du texte pour les graphes
        plt.rc('font', size=18)
```

1 Chargement des données

```
In [ ]: df_test_men = pd.read_csv('Donnees//data_test_men.txt', sep=' ')
        df_train_men = pd.read_csv('Donnees//data_train_men.txt', sep=' ')
        df_test_women = pd.read_csv('Donnees//data_test_women.txt', sep=' ')
        df_train_women = pd.read_csv('Donnees//data_train_women.txt', sep=' ')
```

2 Chargement du modèle de choix discrets

```
In [ ]: parameter_CD = [16.3, 16.5, 12.0, 5.95, 0, -4.16, -7.62, -10.7, -14.3,
                        -17.6, -22.1, -27.2, -31.3, -34.4, -39.5, -50.5, -37.8,
                        -1.05, -0.449, 0.617, 1.63, 2.37, 2.81, 3.14, 3.42, 3.69,
                        3.91, 4.20, 4.50, 4.75, 4.89, 5.14, 5.68, 4.84]
```

```
In [ ]: ASC_V3 = parameter_CD[0]
        ASC_V4 = parameter_CD[1]
        ASC_V5 = parameter_CD[2]
        ASC_V6 = parameter_CD[3]
        ASC_V7 = parameter_CD[4]
        ASC_V8 = parameter_CD[5]
        ASC_V9 = parameter_CD[6]
        ASC_V10 = parameter_CD[7]
```

```
ASC_V11 = parameter_CD[8]
ASC_V12 = parameter_CD[9]
ASC_V13 = parameter_CD[10]
ASC_V14 = parameter_CD[11]
ASC_V15 = parameter_CD[12]
ASC_V16 = parameter_CD[13]
ASC_V17 = parameter_CD[14]
ASC_V18 = parameter_CD[15]
ASC_V19 = parameter_CD[16]
ASC_V20 = 0

B_age_G3 = parameter_CD[17]
B_age_G4 = parameter_CD[18]
B_age_G5 = parameter_CD[19]
B_age_G6 = parameter_CD[20]
B_age_G7 = parameter_CD[21]
B_age_G8 = parameter_CD[22]
B_age_G9 = parameter_CD[23]
B_age_G10 = parameter_CD[24]
B_age_G11 = parameter_CD[25]
B_age_G12 = parameter_CD[26]
B_age_G13 = parameter_CD[27]
B_age_G14 = parameter_CD[28]
B_age_G15 = parameter_CD[29]
B_age_G16 = parameter_CD[30]
B_age_G17 = parameter_CD[31]
B_age_G18 = parameter_CD[32]
B_age_G19 = parameter_CD[33]
B_age_G20 = 0
```

3 Vérification du modèle

In [ ]:
```
start_time = time.clock()
time_j = []

diff_choice = []

for j in range(10):

    cd_choice_test = []

    for index, person in df_test_men.iterrows():

        #Fonctions d'utilité
        V3 = ASC_V3 + B_age_G3 * person.grage
        V4 = ASC_V4 + B_age_G4 * person.grage
        V5 = ASC_V5 + B_age_G5 * person.grage
```

```
        V6 = ASC_V6 + B_age_G6 * person.grage
        V7 = ASC_V7 + B_age_G7 * person.grage
        V8 = ASC_V8 + B_age_G8 * person.grage
        V9 = ASC_V9 + B_age_G9 * person.grage
        V10 = ASC_V10 + B_age_G10 * person.grage
        V11 = ASC_V11 + B_age_G11 * person.grage
        V12 = ASC_V12 + B_age_G12 * person.grage
        V13 = ASC_V13 + B_age_G13 * person.grage
        V14 = ASC_V14 + B_age_G14 * person.grage
        V15 = ASC_V15 + B_age_G15 * person.grage
        V16 = ASC_V16 + B_age_G16 * person.grage
        V17 = ASC_V17 + B_age_G17 * person.grage
        V18 = ASC_V18 + B_age_G18 * person.grage
        V19 = ASC_V19 + B_age_G19 * person.grage
        V20 = 0

        SumV = exp(V3)+exp(V4)+exp(V5)+exp(V6)+exp(V7)+exp(V8)+exp(V9)+exp(V10)
              +exp(V11)+exp(V12)+exp(V13)+exp(V14)+exp(V15)+exp(V16)+exp(V17)
              +exp(V18)+exp(V19)+exp(V20)
        V = [V3, V4, V5, V6, V7, V8, V9, V10, V11, V12, V13, V14, V15, V16,
             V17, V18, V19, V20]


        #Calcul des probabilités
        proba = []

        proba.append(exp(V[0])/SumV)

        for i in range(1, len(V)):
            proba.append((exp(V[i])/SumV)+proba[i-1])

        nbr = random()
        i = 0

        while(nbr>proba[i]):
            i += 1

        cd_choice_test.append(i+3)

    df_test_men['pred_choice'] = cd_choice_test
    tmp_true = plt.hist(df_test_men.grage_C, color='#5472AE',
                        bins=[i for i in range(3, 21)])
    tmp_pred = plt.hist(df_test_men.pred_choice, color='#5472AE',
                        bins=[i for i in range(3, 21)])
    diff_choice.append(sum(abs(tmp_true[0]-tmp_pred[0])))

    time_j.append(time.clock()-start_time)
```

4 Analyse de la distribution de choix

```
In [ ]: df_test_men['pred_choice'] = cd_choice_test
```

```
In [ ]: plt.figure(figsize=(12, 8))
        plt.hist(df_test_men.grage_C-0.1, color='#5472AE', bins='auto')
        plt.hist(df_test_men.pred_choice+0.1, color='#FEA347', bins='auto')
        plt.legend(['Choix réel', 'Choix prédit'])
        plt.xlabel("Catégorie d'âges")
        plt.savefig('Stats//barchart_age_choices.png')
```

```
In [ ]: data = [df_test_men.grage_C, df_test_men.pred_choice]

        plt.figure(figsize=(12, 8))
        bp = plt.boxplot(data)
        setp(bp['boxes'][0], color='#5472AE')
        setp(bp['boxes'][1], color='#FEA347')
        plt.ylabel("Catégorie d'âges")

        xint = range(3, 21)
        plt.yticks(xint)

        #Lignes rouge et bleu temporaires comme aide pour la légende
        hB, = plt.plot([1, 1], '#5472AE')
        hR, = plt.plot([1, 1], '#FEA347')
        plt.legend((hB, hR), ('Choix réel', 'Choix prédit'))
        hB.set_visible(False)
        hR.set_visible(False)

        plt.savefig('Stats//boxplot_age_choices.png')
```

5 Analyse par catégorie d'âges des hommes

```
In [ ]: for i in range(3, 19):

            df = df_test_men[df_test_men.grage==i]

            plt.figure(figsize=(12, 8))
            plt.hist(df.grage_C-0.1, color='#5472AE',
                     bins=np.arange(min(df.grage_C), max(df.grage_C)+0.2, 0.2))
            plt.hist(df.pred_choice+0.1, color='#FEA347',
                     bins=np.arange(min(df.pred_choice), max(df.pred_choice)+0.2, 0.2))
            plt.legend(['Choix réel', 'Choix prédit'])
            plt.title("Hommes de la catégorie d'âges " + str(i))
            plt.xlabel("Catégorie d'âges")
            plt.savefig('Stats//barchart_age_choices_cat'+str(i)+'.png')
```

6 Relation entre les classes d'âges des partenaires

```
In [ ]: df_ages_true = df_test_men.groupby(df_test_men.columns[1:3].tolist()).size()
                                                        .reset_index()
                                                        .rename(columns={0:'Nbr'})
        s = df_ages_true.Nbr
        df_ages_true.plot.scatter('grage', 'grage_C', grid=False, sharex=False,
                                  figsize=(12, 8), s=100, c=s, colormap='Reds')
        xint = range(3, 21)
        plt.yticks(xint)
        plt.xticks(xint)
        plt.xlabel("Catégorie d'âges de l'homme")
        plt.ylabel("Catégorie d'âges de la femme")
        plt.savefig("Stats//scatterplot_ages_true.png")
```

```
In [ ]: df_test_men_pred = df_test_men[['grage', 'pred_choice']]
```

```
In [ ]: df_ages_pred = df_test_men_pred.groupby(df_test_men_pred.columns[0:2].tolist())
                                                                .size().reset_index()
                                                                .rename(columns={0:'Nbr'})
        s = df_ages_pred.Nbr
        df_ages_pred.plot.scatter('grage', 'pred_choice', grid=False, sharex=False,
                                  figsize=(12, 8), s=100, c=s, colormap='Reds')
        xint = range(3, 21)
        plt.yticks(xint)
        plt.xticks(xint)
        plt.xlabel("Catégorie d'âges de l'homme")
        plt.ylabel("Catégorie d'âges de la femme")
        plt.savefig("Stats//scatterplot_ages_pred.png")
        plt.show()
```

7 Analyse des différences d'âges des partenaires

```
In [ ]: plt.figure(figsize=(12, 8))
        plt.hist((df_test_men.grage-df_test_men.grage_C)-0.1, color='#5472AE', bins='auto')
        plt.hist((df_test_men.grage-df_test_men.pred_choice)+0.1, color='#FEA347',
                 bins='auto')
        plt.legend(['Choix réel', 'Choix prédit'])
        plt.xlabel("âge de l'homme - âge de la femme")
        plt.savefig('Stats//barchart_age_differences.png')
```

A.13 Implémentation et vérification du second réseau de neurones

Reseau de neurones pour determiner les preferences des hommes

May 25, 2018

In [1]:
```
#Packages utiles
import itertools
import time
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from random import random
from sklearn.model_selection import GridSearchCV
from sklearn.neural_network import MLPClassifier
from pylab import setp
```

In [2]:
```
#Taille du texte pour les graphes
plt.rc('font', size=18)
```

1 Chargement des données

In [3]:
```
df_test_men = pd.read_csv('Donnees//data_test_men.txt', sep=' ')
df_train_men = pd.read_csv('Donnees//data_train_men.txt', sep=' ')
df_test_women = pd.read_csv('Donnees//data_test_women.txt', sep=' ')
df_train_women = pd.read_csv('Donnees//data_train_women.txt', sep=' ')
```

In [4]:
```
X_train = df_train_men['grage']
y_train = df_train_men['grage_C']
```

In [5]:
```
X_test = df_test_men['grage']
y_test = df_test_men['grage_C']
```

2 Réseau de neurones

In [ ]:
```
start_time = time.clock()

X_train = X_train.reshape(-1, 1)

hidden_layer = [(30,), (25, 25), (40, 40, 40), (60,), (50, 50)]
parameters = {'hidden_layer_sizes': hidden_layer}
```

```
aNN = GridSearchCV(MLPClassifier(solver='adam', activation='relu'),
                   param_grid=parameters, cv=10)
aNN.fit(X_train, y_train)

time_calibr = time.clock()-start_time

X_test = X_test.reshape(-1, 1)
proba_aNN = aNN.predict_proba(X_test)
```

3 Vérification du modèle

In [7]:
```
start_time = time.clock()

diff_dist = []

for j in range(1):

    cd_choice = []

    for index, person in df_test_men.iterrows():

        proba = proba_aNN[index]
        proba = np.cumsum(proba)

        nbr = random()
        i = 0

        while(nbr>proba[i]):
            i += 1

        cd_choice.append(i+3)

    df_test_men['pred_choice'] = cd_choice
    tmp_true = plt.hist(df_test_men.grage_C, color='#5472AE',
                        bins=[k for k in range(3, 21)])
    tmp_pred = plt.hist(df_test_men.pred_choice, color='#5472AE',
                        bins=[k for k in range(3, 21)])
    diff_dist.append(sum(abs(tmp_true[0]-tmp_pred[0])))

time_pred = time.clock()-start_time
```

4 Analyse de la distribution des choix

In [8]:
```
df_test_men['pred_choice'] = cd_choice
```

In [23]:
```
plt.figure(figsize=(12, 8))
```

```
plt.hist(df_test_men.grage_C-0.1, color='#5472AE', bins='auto')
plt.hist(df_test_men.pred_choice+0.1, color='#FEA347', bins='auto')
plt.legend(['Choix réel', 'Choix prédit'])
plt.xlabel("Catégorie d'âges")
plt.savefig('Stats//barchart_age_choices_aNN.png')
```

In [24]:
```
data = [df_test_men.grage_C, df_test_men.pred_choice]

plt.figure(figsize=(12, 8))
bp = plt.boxplot(data)
setp(bp['boxes'][0], color='#5472AE')
setp(bp['boxes'][1], color='#FEA347')
plt.ylabel("Catégorie d'âges")

xint = range(3, 21)
plt.yticks(xint)

#Lignes bleue et rouge temporaires pour créer la légende
hB, = plt.plot([1, 1], '#5472AE')
hR, = plt.plot([1, 1], '#FEA347')
plt.legend((hB, hR), ('Choix réel', 'Choix prédit'))
hB.set_visible(False)
hR.set_visible(False)

plt.savefig('Stats//boxplot_age_choices_aNN.png')
```

5 Analyse par catégorie d'âges des hommes

In [25]:
```
for i in range(3, 19):

    df = df_test_men[df_test_men.grage==i]

    plt.figure(figsize=(12, 8))
    plt.hist(df.grage_C-0.1, color='#5472AE',
             bins=np.arange(min(df.grage_C), max(df.grage_C)+0.2, 0.2))
    plt.hist(df.pred_choice+0.1, color='#FEA347',
             bins=np.arange(min(df.pred_choice), max(df.pred_choice)+0.2, 0.2))
    plt.legend(['Choix réel', 'Choix prédit'])
    plt.title("Hommes de la catégorie d'âges "+str(i))
    plt.xlabel("Catégorie d'âges")
    plt.savefig('Stats//barchart_age_choices_cat'+str(i)+'_aNN.png')
```

6 Relation entre la catégorie d'âges des hommes et celle des femmes

In [ ]:
```
df_ages_true = df_test_men.groupby(df_test_men.columns[1:3].tolist()).size()
                                                    .reset_index()
                                                    .rename(columns={0:'Nbr'})
```

```
        s = df_ages_true.Nbr
        df_ages_true.plot.scatter('grage', 'grage_C', grid=False, sharex=False,
                                  figsize=(12, 8), s=100, c=s, colormap='Reds')
        xint = range(3, 21)
        plt.yticks(xint)
        plt.xticks(xint)
        plt.xlabel("Catégorie d'âges de l'homme")
        plt.ylabel("Catégorie d'âges de la femme")
        plt.savefig("Stats//scatterplot_ages_true_aNN.png")
```

```
In [14]: df_test_men_pred = df_test_men[['grage', 'pred_choice']]
```

```
In [ ]: df_ages_pred = df_test_men_pred.groupby(df_test_men_pred.columns[0:2].tolist())
                                                                .size().reset_index()
                                                                .rename(columns={0:'Nbr'})
        s = df_ages_pred.Nbr
        df_ages_pred.plot.scatter('grage', 'pred_choice', grid=False, sharex=False,
                                  figsize=(12, 8), s=100, c=s, colormap='Reds')
        xint = range(3, 21)
        plt.yticks(xint)
        plt.xticks(xint)
        plt.xlabel("Catégorie d'âges de l'homme")
        plt.ylabel("Catégorie d'âges de la femme")
        plt.savefig("Stats//scatterplot_ages_pred_aNN.png")
```

7 Age differences analysis

```
In [33]: plt.figure(figsize=(12, 8))
         plt.hist((df_test_men.grage-df_test_men.grage_C)-0.1, color='#5472AE', bins='auto')
         plt.hist((df_test_men.grage-df_test_men.pred_choice)+0.1, color='#FEA347',
                  bins='auto')
         plt.legend(['Choix réel', 'Choix prédit'])
         plt.xlabel("âge de l'homme - âge de la femme")
         plt.savefig('Stats//barchart_age_differences_aNN.png')
```

A.14 Premier modèle probabiliste pour effectuer le couplage

Premier modele pour coupler les individus du marche

May 19, 2018

In [8]:
```
#Packages utiles
import itertools
import time
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from random import random
from sklearn.model_selection import GridSearchCV
from sklearn.neural_network import MLPClassifier
from pylab import setp
```

In [3]:
```
#Taille du texte pour les graphes
plt.rc('font', size=18)
```

1 Chargement des données

In [4]:
```
df_test_men = pd.read_csv('Donnees//data_test_men.txt', sep=' ')
df_train_men = pd.read_csv('Donnees//data_train_men.txt', sep=' ')
df_test_women = pd.read_csv('Donnees//data_test_women.txt', sep=' ')
df_train_women = pd.read_csv('Donnees//data_train_women.txt', sep=' ')
```

In [5]:
```
X_train = df_train_men['grage']
y_train = df_train_men['grage_C']
```

In [6]:
```
X_test = df_test_men['grage']
y_test = df_test_men['grage_C']
```

2 Chargement du réseau de neurones

In [7]:
```
X_train = X_train.reshape(-1, 1)

hidden_layer = [(30,), (25, 25), (40, 40, 40), (60,), (50, 50)]
parameters = {'hidden_layer_sizes': hidden_layer}

aNN = GridSearchCV(MLPClassifier(solver='adam', activation='relu'),
                   param_grid=parameters, cv=10)
```

```
aNN.fit(X_train, y_train)

X_test = X_test.reshape(-1, 1)

proba_aNN = aNN.predict_proba(X_test)
```

3 Calcul des préférences

```
In [17]: def compute_choices(index):

             proba = proba_aNN[index]
             proba = np.cumsum(proba)
             nbr = random()
             i = 0

             while(nbr>proba[i]):
                 i += 1

             return i+3
```

4 Réalisation du couplage

```
In [ ]: start_time = time.clock()
        nbr_marriage = np.zeros(10)

        for j in range(10):

            #Disponibilité des femmes par catégorie d'âges
            df_women_age = df_test_women['grage'].value_counts()
            df_women_age[19] = 0

            cat = [i for i in range(3, 20)]
            data = np.zeros(len(cat), dtype=int)

            nbr_single = pd.Series(data=data, index=cat)

            for index, person in df_test_men.iterrows():

                choice = compute_choices(index)

                if(df_women_age[choice]>0):

                    nbr_marriage[j] += 1
                    df_women_age[choice] -= 1

                else:
```

```
                nbr_single[person.grage] += 1

        time_execution = time.clock()-start_time
```

5 Analyse des célibataires restant dans le marché

In [ ]:
```
#Histogramme des hommes
plt.figure(figsize=(12, 8))
plt.bar(nbr_single.index, nbr_single.values, color='#5472AE')
plt.xlabel("Catégorie d'âges")
xint = range(3, 21)
plt.xticks(xint)
plt.savefig('Stats//barchart_age_single_M_M1.png')
```

In [ ]:
```
#Histogramme des femmes
plt.figure(figsize=(12, 8))
plt.bar(df_women_age.index, df_women_age.values, color='#5472AE')
plt.xlabel("Catégorie d'âges")
xint = range(3, 21)
plt.xticks(xint)
plt.savefig('Stats//barchart_age_single_F_M1.png')
```

A.15 Méthode du recuit simulé réalisant le couplage des individus

Algorithme du recuit simule pour le couplage des individus du marche

May 25, 2018

In [1]:
```
#Packages utiles
import itertools
import time
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from random import randint
from random import random
from sklearn.model_selection import GridSearchCV
from sklearn.neural_network import MLPClassifier
from pylab import setp
```

In [3]:
```
#Taille du texte pour les graphes
plt.rc('font', size=18)
```

1 Chargement des données

In [4]:
```
df_test_men = pd.read_csv('Donnees//data_test_men.txt', sep=' ')
df_train_men = pd.read_csv('Donnees//data_train_men.txt', sep=' ')
df_test_women = pd.read_csv('Donnees//data_test_women.txt', sep=' ')
df_train_women = pd.read_csv('Donnees//data_train_women.txt', sep=' ')
```

In [5]:
```
X_train = df_train_men['grage']
y_train = df_train_men['grage_C']
```

In [6]:
```
X_test = df_test_men['grage']
y_test = df_test_men['grage_C']
```

2 Chargement du réseau de neurones

In [7]:
```
X_train = X_train.reshape(-1, 1)

hidden_layer = [(30,), (25, 25), (40, 40, 40), (60,), (50, 50)]
parameters = {'hidden_layer_sizes': hidden_layer}
```

```
aNN = GridSearchCV(MLPClassifier(solver='adam', activation='relu'),
                   param_grid=parameters, cv=10)
aNN.fit(X_train, y_train)

X_test = X_test.reshape(-1, 1)

proba_aNN = aNN.predict_proba(X_test)
```

3 Simulation du couplage

In [8]:
```
#Fonctions nécessaires à la résolution du recuit simulé

def function_calculate(matching, probabilities):
    '''
    Fonction calculant la somme des probabilités associées à chaque couple.
    '''

    sum_proba = 0
    i = 0

    for age in matching:

        proba_man = probabilities[i]

        sum_proba = sum_proba + proba_man[age-3]

        i += 1

    return sum_proba


def neighbour_calculate(matching):
    '''
    Fonction déterminant un voisin du couplage entré.
    '''

    matching_N = list(matching)

    nbr1 = randint(0, len(matching_N)-1)
    nbr2 = randint(0, len(matching_N)-1)
    matching_N[nbr1], matching_N[nbr2] = matching_N[nbr2], matching_N[nbr1]

    return matching_N


def CritMetropolis(delta, T):
    '''
```

```
        Fonction aidant à choisir le couplage suivant dans notre algorithme.
        '''

        nbr = random()

        if(delta<=0):
            return True
        elif(nbr<np.exp(-delta/T)):
            return True
        else:
            return False
```

In [ ]:
```
#Liste des femmes disponibles
women_age_av = df_test_women['grage'].values

#Mélange aléatoire de la liste
np.random.shuffle(women_age_av)

start_time = time.clock()
time_i = []

#Solution initiale
matching = list(women_age_av)

#Température initiale
T = 15

#Itération maximale
iter_max = 50000
iter_i = 0
#Nombre d'itérations par palier
iter_palier = 50

val1 = function_calculate(matching, proba_aNN)
val2 = 0
function_g = []

#Initialisation du taux d'acceptation
acceptance_rate = 1
#Borne de ce taux
epsilon = 0.1

coef = 0.9

while(iter_i<iter_max and acceptance_rate>epsilon):

    nb_moves = 0
    function_g.append(val1)
```

```
        for i in range(iter_palier):

            #Voisin aléatoire du matching
            matching_N = neighbour_calculate(matching)

            val2 = function_calculate(matching_N, proba_aNN)

            delta = val1-val2

            if(CritMetropolis(delta, T)):
                matching = list(matching_N)
                val1 = val2
                nb_moves += 1

        if(nb_moves==0):
            acceptance_rate = 0
        else:
            acceptance_rate = nb_moves/iter_palier

        #Mise à jour de T
        T = T*coef

        iter_i += 1

        time_i.append(time.clock()-start_time)
```

4 Analyse de la croissance de la fonction objectif par itération

In [ ]:
```
plt.figure(figsize=(14, 8))

x = [i for i in range(iter_i)]
z = [function_calculate(df_test_men.grage_C.values, proba_aNN) for i in range(iter_i)]

plt.plot(x, function_g, '-', linewidth=2)

plt.plot(x, z, '-', linewidth=2)

plt.grid(True)
plt.legend(['Valeur du couplage obtenu', 'Valeur du couplage réel'])
plt.xlabel('Numéro de simulation')
plt.ylabel('Valeur de la fonction objectif')
plt.savefig('Stats//line_fct_obj_recuit.png')
```

5 Analyse des mariages engendrés

```
In [ ]: df_ages_true = df_test_men.groupby(df_test_men.columns[1:3].tolist())
                                                        .size().reset_index()
                                                        .rename(columns={0:'Nbr'})
        s = df_ages_true.Nbr
        df_ages_true.plot.scatter('grage', 'grage_C', grid=False, sharex=False,
                                  figsize=(12, 8), s=100, c=s, colormap='Reds')
        xint = range(3, 21)
        plt.yticks(xint)
        plt.xticks(xint)
        plt.xlabel("Catégorie d'âges de l'homme")
        plt.ylabel("Catégorie d'âges de la femme")
        plt.savefig("Stats//scatterplot_ages_true_recuit.png")

In [27]: df_test_men['pred_choice'] = matching

In [28]: df = df_test_men[['grage', 'pred_choice']]

         df_bis = df.groupby(df.columns[0:2].tolist()).size().reset_index()
                                .rename(columns={0:'Nbr'})

In [ ]: s = df_bis.Nbr
        df_bis.plot.scatter('grage', 'pred_choice', grid=False, sharex=False,
                                  figsize=(12, 8), s=100, c=s, colormap='Reds')
        xint = range(3, 21)
        plt.yticks(xint)
        plt.xticks(xint)
        plt.xlabel("Catégorie d'âges de l'homme")
        plt.ylabel("Catégorie d'âges de la femme")
        plt.savefig("Stats//scatterplot_ages_pred_recuit.png")

In [ ]: plt.figure(figsize=(12, 8))
        plt.hist((df_test_men.grage-df_test_men.grage_C)-0.1, color='#5472AE',
                 bins=np.arange(min(df_test_men.grage-df_test_men.grage_C),
                                max(df_test_men.grage-df_test_men.grage_C)+0.2, 0.2))
        plt.hist((df['grage']-df['pred_choice'])+0.1, color='#FEA347',
                 bins = np.arange(min((df['grage']-df['pred_choice'])),
                                  max((df['grage']-df['pred_choice']))+0.2, 0.2))
        plt.legend(['Choix réel', 'Choix prédit'])
        plt.xlabel("âge de l'homme - âge de la femme")
        plt.savefig('Stats//marriage_age_differences_recuit.png')
```